The Effects of Feature Geometry on Simulating Nanoimprint Lithography

by

Cai P. GoGwilt

B.S., Massachusetts Institute of Technology (2010)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

Author .....
Department of Electrical Engineering and Computer Science
May 18, 2011

Certified by .....
Duane S. Boning
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Christopher J. Terman
Chairman, Department Committee on Graduate Theses

The Effects of Feature Geometry on Simulating Nanoimprint Lithography

by

Cai P. GoGwilt

Submitted to the Department of Electrical Engineering and Computer Science
on May 18, 2011, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Nanoimprint lithography (NIL) is a method for fabricating nano-scale patterns by pressing stamps into viscous materials. A key barrier to industry adoption of NIL is the inability to predict whether a stamp will imprint successfully and how long the process should be run for. In this thesis, we help quantify the accuracy loss for an existing simulation package, `simprint`, which supports geometric abstractions and can simulate at the die level. To do this, we develop and study several comparison metrics. Our temporal submetric quantifies the error between two simulations at each timestep, while our spatial submetric quantifies the error at each spatial location. We subsequently use these metrics to study pattern abstraction by looking at how different types of patterns lead to different errors. This would allow us to suggest pattern abstractions that could improve the accuracy of a simulation. However, none of the features we study correlate with error. We conclude by exploring other possible uses of our metrics.

Thesis Supervisor: Duane S. Boning
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I could not have completed this thesis without the incredible guidance and support of my friends and colleagues. In particular, I would like to thank Hayden Taylor for all the time and effort he spent instructing me. Fundamentally, without his work on `simprint`, this thesis could not have happened. Furthermore, without his advice and attention, I would not have known how to proceed. I would also like to thank Duane Boning for helping to guide my research in addition to guiding my academic career at MIT. I would like to thank MTL's Statistical Metrology group for being so welcoming to me, and for noticing interesting components of my research that I would not have addressed had they not brought them to my attention. Thank you to my family, especially my parents and my brother, for their support and love. Finally, thank you to my friends, and in particular to Jessica Ho for her support and encouragement, and for patiently listening to me monologue incomprehesibly about metrics for models, data structures for VLSI, and simulations for NIL.

Contents

1 Introduction **13**

1.1 Motivation 14

1.2 Problem 15

1.3 Approach 15

1.4 Contribution 15

1.5 Organization of Thesis 16

2 Background **17**

2.1 NIL and Process Variants 18

2.1.1 Thermal Nanoimprint Lithography (TNIL) 19

2.1.2 UV Nanoimprint Lithography (UV-NIL) 20

2.1.3 Comparison of TNIL and UV-NIL 21

2.2 Models of Nanoimprinting 21

2.2.1 Models of TNIL 22

2.2.2 Models of UV-NIL 23

2.2.3 Hierarchical Models 23

2.3 `Simprint` 24

2.4 Implications and Considerations 24

3 Metrics **27**

3.1 Models and Metrics 28

3.1.1 Compatible Models 28

3.1.2 Metrics 29

3.2 Characterizing RLT Difference 30
3.2.1 Characteristics of Square Error 30
3.2.2 Characterizing Noise and Decay 32
3.3 Temporal Submetric 33
3.3.1 Behavior of Temporal Submetric 34
3.3.2 Temporal Submetric as Variance 34
3.4 Spatial Submetric 36
3.4.1 Behavior of Spatial Submetric 38
3.4.2 Spatial Submetric as Variance and Noise 39
3.5 Unscaled Metric 41
3.6 Metrics and Error 42
3.7 Other Metric Candidates 45
3.7.1 Temporal Submetric 45
3.7.2 Spatial Submetric 45
3.8 Conclusion 46

4 The Effects of Feature Abstraction **49**
4.1 Background 50
4.2 Extending Metrics to the Discrete Space 50
4.3 Method 51
4.3.1 Test Stamps 52
4.3.2 Simulations 52
4.4 Results 52
4.4.1 Effect of Abstraction Level on Accuracy 54
4.4.2 Effect of Features on Accuracy 54
4.5 Discussion 56

5 Conclusion **63**
5.1 Metrics 64
5.1.1 Applications to Research 64
5.1.2 Applications to Industry 64

5.2 Metric Limitations 64
5.3 `Simprint`'s Reliability in Hierarchical Mode 65
5.4 Hierarchical Pattern Exploration 65

List of Figures

1-1 Adding simulation to the NIL manufacturing process. 14

2-1 The imprinting process of NIL. 18

2-2 Example simulation results from `simprint`. 25

3-1 The residual layer thickness. 27

3-2 The RLT predicted by the simulation of a test chip being imprinted by a typical T-NIL process. 31

3-3 An idealized illustration of the RLT square error curve. 32

3-4 A log-log plot of the time-evolution of the square RLT difference. . . 33

3-5 The time evolution of the temporal submetric, ξ_t. 35

3-6 A plot of how the range in error values and density of error distribution change with time. 36

3-7 Quantile plots of absolute error and ξ_t and the time-varying proportion of error values below ξ_t. 37

3-8 Intensity map of the spatial submetric. 38

3-9 Log log plots of the difference in RLT time evolutions with corresponding values of ξ_s. 39

3-10 The distribution of error values at several points on the pattern. . . . 40

3-11 A distribution plot of the proportion of error values below $\xi_s(x, y)$ for every grid region (x, y) in the pattern. 41

3-12 Plots showing how the unscaled metric and submetrics characterize error. 43

3-13 Using the metrics as measures of error. 44

4-1 Test patterns and the RLTs predicted by flat and hierarchical models. 53
4-2 The ξ_t curves produced by the 50× hierarchical simulations. 55
4-3 ξ_s for each patch plotted against the abstraction level of the hierarchical simulation. 57
4-4 Effect of density on spatial error metric. 57
4-5 Effect of perimeter on spatial error metric. 58
4-6 Effect of perimeter ratio on spatial error metric. 58
4-7 Effect of number of polygons on spatial error metric. 59
4-8 Effect of feature size on spatial error metric. 59
4-9 Sensitivity of pattern density to time-step size. 61

Chapter 1

Introduction

Nanoimprint lithography (NIL) is a method for fabricating nano-scale patterns by pressing stamps into viscous materials. It appears to have great potential for manufacturing semiconductor, photonic and MEMS (microelectromechanical systems) structures, and is not subject to the properties of diffraction that limit optical lithography, the dominant nanofabrication technique [12, 2]. Several models and simulations of the imprinting process have been developed, but a key barrier to industry adoption of NIL remains the inability to efficiently simulate the process at die or wafer level [12].

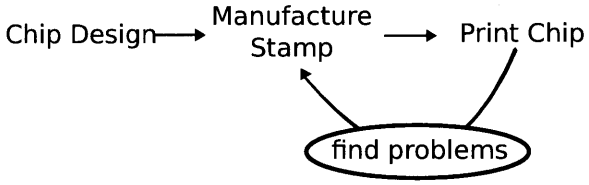
To simulate NIL at the die or wafer level, hierarchical approximations of feature-level models need to be developed. At the Microsystems Technology Laboratories, Hayden Taylor has developed `simprint`, a simulation tool based on his model of the NIL process. It approximates large blocks in the feature geometries as basic patterns, and combines precomputed results for these blocks [14].

In this thesis, we (1) develop a method for model comparison and (2) use this method to test simprint's hierarchical simulation mode. The comparison method is useful for evaluating a wide range of NIL models, and quantifies the accuracy of hierarchical simulations. It also provides a system for improving the hierarchical simulations by studying the effects of feature geometry. Studying how feature geometry affects the stability of NIL simulations may also guide design rules and fill patterns that allow efficient and reliable imprinting.

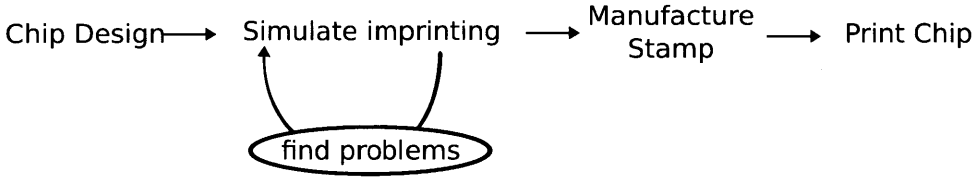
1.1 Motivation

The purpose of this study is to assist the use of NIL for industrial nanofabrication. In particular, we hope that a good understanding of NIL simulation accuracy and a fast enough simulation will assist in the efficient and accurate use of NIL. One possible set of steps for NIL is (1) designing the stamp, (2) producing the stamp and (3) printing with the stamp. Simulation would be an extra step between steps (1) and (2), so to justify its use, it would need to be cost- and time-effective compared to the other steps, as well as provide some benefit to the other steps.

The benefit of simulation is that it allows us to quickly and cheaply modify our stamp design to make the NIL process faster and more stable. Without this added step, the stamp used for NIL might need to be imprinted for longer, or imprint imperfect patterns in critical regions. Figure 1-1 illustrates how simulation would fit into the manufacturing process and potentially help.



(a) Current process.



(b) Proposed process.

Figure 1-1: Adding simulation to the NIL manufacturing process.

The simulation efficiency remains a problem, though. Conceivably, we could build very accurate finite element models (FEMs), but the computing power and simulation time necessary to do this would cost more in time or expense than manufacturing a stamp and testing it. `Simprint` helps this issue by simulating at a hierarchical level. However, simulations continue to be prohibitively expensive in time and memory for real chips without further approximations.

1.2 Problem

We can generally address the problem of simulation efficiency by decreasing granularity. We typically do this by either decreasing the number of time-steps it takes a simulation to complete or increasing the length scale it simulates over. For `simprint`, decreasing the number of time-steps is straightforward. `Simprint` can also increase the simulation length scale, but this requires some way of abstracting blocks of patterns. This is the approach we will focus on in this thesis.

For simulation, the trade-off with efficiency is often accuracy. We can measure the efficiency of a simulation method by measuring the time it takes to run, but it is more difficult to measure the loss of accuracy. Quantifying how close two model predictions are allows us to do this.

1.3 Approach

In order to quantify the loss of accuracy in `simprint`, we will develop a set of metrics for comparison. We will then explore the properties of these metrics and relate them to important features of the simulations they compare. Ultimately, we will apply these metrics to `simprint`'s pattern abstraction and search for pattern features that lead to simulation instability.

The comparison metrics we develop will need to correspond to values that we care about when using NIL. For example, we will care more about the later times in a simulation, since we are unlikely to stop a stamp's imprinting early in the process. The metrics should therefore forgive differences that occur early in the process.

1.4 Contribution

In this thesis, we develop and study a set of comparison metrics, and use these to measure the accuracy of hierarchical approximations compared to the unabstracted "flat" simulations. We also use the comparison metrics to search for pattern features that caused simulation instability. The metrics are developed to be general-purpose.

Though they are only applied to `simprint` here, we intend them to apply to other simulations of NIL and even to experiments. Our application of comparison metrics to studying how feature abstraction affects accuracy gives us even more confidence in `simprint`'s hierarchical mode. It also guides how we should choose the scale of abstraction to use. Finally, we suggest how to apply these metrics to improving hierarchical methods of simulation.

1.5 Organization of Thesis

We review the range of techniques that NIL encompasses in chapter 2. These techniques can be divided into thermal nanoimprint lithography (TNIL) and UV nanoimprint lithography (UV-NIL) based on whether they use cooling or UV-curing to solidify the imprinted material. We discuss work on simulating both. We then focus on `simprint`, the simulation package we studied. In chapter 3, we study how two simulations of TNIL can diverge. We then use observations from this study to develop comparison metrics, and suggest applications for these metrics. Chapter 4 sets up the problem of simulating at a hierarchical level, and describes how to study stamp topographies with the metrics from chapter 3. We then study several features using this method. We conclude in chapter 5 by describing open research questions and suggesting new applications for comparison metrics.

Chapter 2

Background

The term "nanoimprint lithography" (NIL) refers to a wide range of processes for nano-fabrication. It is being developed in parallel to compete with or complement the existing dominant lithographic technique, optical or photolithography (PL), for industrial use. Though NIL is currently slower and therefore less cost-effective, PL is limited by the properties of diffraction [2, 1, 6, 14]. Because NIL is in development, it is unclear which of several process variants will become standard. Additionally, the costs and benefits of NIL must compare favorably (or at least competitively) with PL to gain industry acceptance.

NIL processes vary in many ways, including material choice, resist configuration, stamp orientation, and hardening mechanism. Even the imprinting process itself is significantly affected by the particular NIL method. For example, a resist may be evenly coated onto a substrate, or sprayed onto the substrate to form droplets. In the first case, elastic deformation and viscous flow may dominate imprinting [14, 11], while in the second case, capillary forces and air bubble diffusion may be more important [10].

Understanding the imprinting process is crucial to the efficiency of NIL, since the imprinting process contributes substantially to (and often dominates) the total process time. Several very good and compelling models of the NIL process exist. Many of these models focus on simulating one or two patterned features, or a very regular array of features [1, 10, 11, 6]. Others focus on simulating the imprinting of

a full die [14, 13, 7]. `Simprint`, the simulation tool we study here, falls in the latter category [14].

Modeling the imprinting of a full chip is important for making NIL viable for industry. In NIL, the stamp is often held for a longer time than necessary to avoid bad imprinting and damaged stamps. A good simulation would allow manufacturers to be less conservative, and save fabrication time.

In this chapter, we explain the motivation for studying and improving hierarchical simulations of NIL, discuss the features of existing NIL simulations, and mention some of the main factors in modeling NIL. Section 2.1 describes several existing process variants of NIL, and briefly discusses the features of PL. Section 2.2 outlines work on modeling NIL. In section 2.3, we focus on `simprint`, the model we studied. Finally, we conclude by discussing the concerns with and solutions to bringing NIL to industry.

2.1 NIL and Process Variants

NIL describes a wide range of processes for imprinting material at a small scale. In NIL, a patterned stamp is pressed into a viscous resist in order to transfer the relief of the pattern to the material (see figure 2-1). The resist is then hardened in some way and the stamp is removed and reused. In lithographic applications, this resist layer is coated onto a substrate that is being patterned. It is then used as an etch mask and subsequently removed, leaving the pattern on the material underneath. However, NIL can be used directly as a non-lithographic technique, so this etch step is not always necessary [2, 4].

Figure 2-1: The imprinting process of NIL.

Many different methods of NIL have been developed, and in the following sec-

tions, we separate them into two categories (using the categorization of [2]): thermal nanoimprint lithography (TNIL) and UV nanoimprint lithography (UV-NIL). In TNIL, a thin thermoplastic film is imprinted and hardened by cooling, while in UV-NIL, a liquid resist of droplets is solidified by UV curing [12]. Within these two classes, many NIL processes exist. We will discuss some of the details in the following sections, which describe the particulars of TNIL and UV-NIL.

2.1.1 Thermal Nanoimprint Lithography (TNIL)

We use the term TNIL to refer to a large class of NIL processes that solidify a resist by cooling it below its glass transition temperature (T_g). In older literature, it has also been referred to as hot embossing lithography (HEL) [12]. For our purposes, TNIL includes step and stamp imprint lithography (SSIL), room temperature NIL (RT-NIL), and Obducat's simultaneous combined thermal and UV NIL (STU-NIL) [12].

In TNIL, the stamp is typically made out of a hard and durable material such as silicon or silicon dioxide. It is also important that the stamp's thermal expansion matches that of the substrate, so silicon stamps are often paired well with silicon substrates [4]. The stamps themselves are manufactured using an alternative lithographic technique (such as electron beam lithography) and reactive ion etching [3, 4].

The stamp is then pressed into a layer of viscous resist. The resist is typically heated to about 70° C above its T_g to encourage viscous flow [4, 12]. The resist is then imprinted, followed by cooling and demolding at about 20° C below T_g [12]. A typical choice for resist is polymethyl methacrylate (PMMA) or polystyrene (PS), especially since they come in a range of molecular weights and can be used for a range of thicknesses [12]. These may be mixed with polydimethylsiloxane (PDMS) to promote stamp release [4].

Older imprinting processes used stiff hydraulic, air, or screw mechanisms, which could result in uneven pressure distributions across the imprinted stamp. Many TNIL systems now use air-pressurized membranes [12]. SSIL uses a smaller stamp to "step" across the wafer, but uses the same cooling mechanism. Obducat uses TNIL to emboss

hydrophobic intermediate polymer stamps (IPS), which then imprint resists using STU-NIL. This allows them to wash the original stamp to prevent contamination [12].

2.1.2 UV Nanoimprint Lithography (UV-NIL)

In UV-NIL, the resist is a UV-curable, low viscosity liquid, which is imprinted and then solidified by exposing it to UV light [12, 10]. Occasionally, UV-NIL is referred to as photo nanoimprint lithography (P-NIL), photo imprint lithography (PIL), or soft lithography (due to the soft stamp) [2, 3]. Step and flash imprint lithography (SFIL) uses the same stepping idea as SSIL, but hardens the resist using UV-curing [12]. Roll-to-roll and roll-to-plate nanoimprint lithography (R2RNIL/R2PNIL) also use UV-curable resists, but the stamp is attached to rollers like a conveyer belt and the resist is squeezed through, cured, and released as it travels with the stamp [1].

UV-NIL stamps must be UV-transparent. However, since the process requires much less pressure than TNIL, elastomeric materials (such as PDMS) can also work as stamps [12]. Softer materials allow the stamp to conform to the substrate and resist better, but may not last as long [2].

Resists must be UV-curable, and are typically low-viscosity to assist in imprinting. Unlike in TNIL, where the resist is coated on as a thin layer, in UV-NIL, resists are typically sprayed on with droplets as small as $5\mu m$. The density of the spray can be optimized to fit the stamp and make imprinting faster [12, 3]. UV-NIL resists are often proprietary, but Costner et al. recently described the resist used by Molecular Imprints, Inc. in [3]. Here, they mention that the resist "typically consists of a bulk polymerizable organic monomer, such as an acrylate or vinyl ether (VE), a silicon-containing or siloxane-containing monomer to provide oxygen-etch resistance, a cross-linking agent to provide mechanical strength and thermal stability to the imprint structure, a photoinitiator, and a fluorinated surfactant to promote template release" [3].

The imprinting method is usually very similar to TNIL. One interesting modification of this method is R2R/R2P-NIL. In this method, the stamp is rolled onto a resist

on either a flexible substrate or a rigid plate through a series of conveyer belts. Ahn et al. have demonstrated 4-inch wide continuous imprinting of nano-gratings using this method [1].

2.1.3 Comparison of TNIL and UV-NIL

TNIL and UV-NIL both have comparative benefits and drawbacks to one another. TNIL has been studied for longer, and TNIL resists can be tweaked and varied to satisfy particular properties [12]. However, UV-NIL is not limited to imprinting viscous liquids near their glass transition point, and can therefore use liquids that flow much more easily. Because UV-NIL resists are often liquids, UV-NIL requires much less pressure. Subsequently, stamps can be a lot less stiff [3]. These flexible stamps can be peeled off so as to encourage release between the stamp and the resist. However, softer stamps are also less durable, so they do not last as long and are susceptible to defects [12].

For the purposes of modeling the imprinting process, TNIL and UV-NIL are somewhat different due to the difference in resist properties. In TNIL, the resists are viscous liquids close to T_g. Thus, it takes a significant amount of time to get the resist to conform to the stamp. TNIL models typically focus on the residual layer thickness (RLT) – how much resist is left between stamp and substrate – since RLT variation hurts etching and limits the process [11]. In UV-NIL, liquid resists reduce RLT variation quickly and allow for potentially faster imprints. Spraying resists onto a substrate can make imprinting easier. However, UV-NIL techniques must wait for droplets to merge and trapped gas bubbles to disappear [10]. We elaborate on the specifics of modeling NIL techniques in the next section.

2.2 Models of Nanoimprinting

Properly simulating the imprinting process of NIL is one key to having NIL adopted by industry. Simulations of NIL typically simulate a very small area of pattern. This is useful for understanding the feature dependent limitations of NIL and developing

design rules [11, 10]. However, for dies and wafers, these models are limited to very regular patterns. Hierarchical simulations (section 2.2.3) model NIL at a level where feature-rich patterns are tractable. With these simulations, chip designers could actually iterate on their designs to optimize them using the simulation [13].

The difference in resist and stamp properties between TNIL and UV-NIL also divides NIL models into those suitable for modeling one or the other. As mentioned before, TNIL models tend to focus on predicting the RLT uniformity, since the high viscosity of TNIL resists forces the process to take longer as the resist flows slowly to less dense regions [7]. Models of UV-NIL, on the other hand, tend to focus on how air bubbles combine and dissolve, since the resist can perform poorly in the etching step if trapped air bubbles are too large in size or number [10]. Additionally, since UV-NIL resists are much less viscous, models of UV-NIL must take capillary forces into account [10].

In the rest of this section, we will review existing models of TNIL and UV-NIL. We will then discuss hierarchical methods, which are capable of analyzing larger pattern areas.

2.2.1 Models of TNIL

Several finite element models (FEMs) of the TNIL imprinting process exist. They usually model the resist as a viscous liquid or non-linear elastic material conforming to a rigid stamp [11]. Using FEMs, researchers have been able to explore the importance of various physical properties of NIL stamps and resists. They have also managed to achieve good fits with experiment [14]. Several main factors have been considered for FEMs. These include effects such as shear deformation, viscous flow, and capillary flow [11].

Molecular dynamics (MD) models typically require more computation than FEMs to simulate the same amount of patterning. However, they can capture behavior that FEMs have difficulty with. MD models of TNIL that factor in adhesion and friction forces due to process conditions have helped to study how resists can tear and deform as the stamp is removed [6].

Though FEMs and MD models can model NIL quite well, they are too slow for irregular pattern simulation at the die or wafer scale. Many FEM simulations have been done on uniform repeating geometries [11]. However, some applications of NIL, such as chip manufacturing, require irregular stamps of a much larger scale. FEMs are not feasible in this case, due to the very large number of features that would require simultaneous simulation.

2.2.2 Models of UV-NIL

Models of UV-NIL tend to be concerned with how droplets of resist merge and how air bubbles dissolve. Because the resist is liquid, capillary forces cannot be ignored (as they are in most TNIL models). Analytical models of droplets merging and bubble trapping [10], air bubble dissolving and shrinkage [8], and UV exposure and curing [5] have been explored. An analytical droplet merging study using lubrication theory by Reddy et al. showed that UV-NIL's imprint time decreases with the number of droplets [10]. Reference [5] suggests that analytical models could be used for the optimization of process conditions and material properties.

2.2.3 Hierarchical Models

For simulations to be used in practice for feature-rich chip patterns, NIL simulations must make approximations about the models they are based on and the stamps they simulate. The resulting simulations tend to be less accurate than their more fine-grained FEM or MD counterparts. However, they can simulate larger patterns in reasonable amounts of time. This will allow manufacturers to optimize stamp patterns, process conditions, and material properties before production [7].

Methods for simulating NIL on complex patterns tend to divide the pattern into a coarse grid. In these hierarchical simulations, each cell in the coarse grid represents many features as a single unit, often only keeping track of key feature statistics such as pattern density or feature scale. These statistics may then be used to approximate each cell of complex patterns as a regular array of simple shapes (see figure 2-2

for an example of simulation results using a hierarchical model). The simulations then track values such as pressure distribution and RLT for each cell at this coarse discretization [14, 7]. Taylor's method for quickly simulating complex patterns is implemented in `simprint`. This program can simulate feature-rich patterns at both feature and hierarchical levels quickly [14].

2.3 Simprint

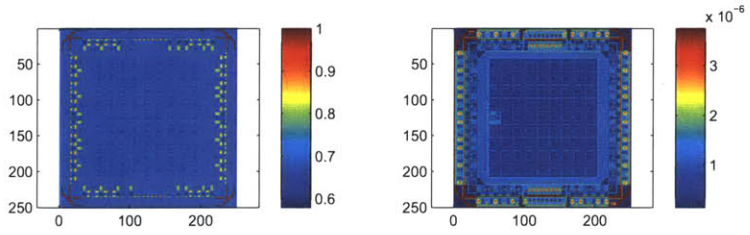
Throughout this thesis, we will focus on Taylor's program for modeling NIL, `simprint`. This is described in detail in Taylor's PhD thesis [14]. In this section, we will briefly review some of the features of `simprint`.

`Simprint` enables users to simulate the imprinting of a stamp topography under NIL (and microembossing) conditions. Figure 2-2 shows an example of the results of a typical simulation. Stamp and substrate parameters such as material and thickness can be specified and simulated under different process conditions, including temperature, pressure, and hold-time. The results are estimates of RLTs, pressure distributions, and cavity fill ratios for several time-steps, as illustrated in figure 2-2. Simulation parameters such as number of time-steps and convergence conditions can also be set [14].

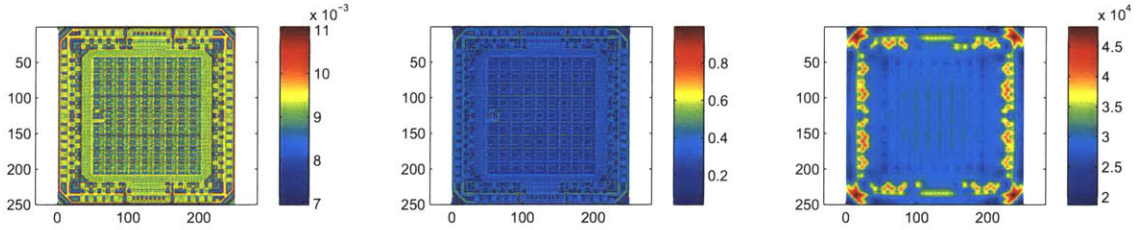
`Simprint` models the imprinted material as a viscoelastic solid and applies a thin-film impulse function based on [9]. Additionally, in hierarchical mode, it accelerates the simulation by approximating large patches of the stamp as regular geometries such as lines, square holes, or square protrusions. This hierarchical simulation mode uses precomputed material responses to these patterns to quickly simulate large areas of the pattern [14].

2.4 Implications and Considerations

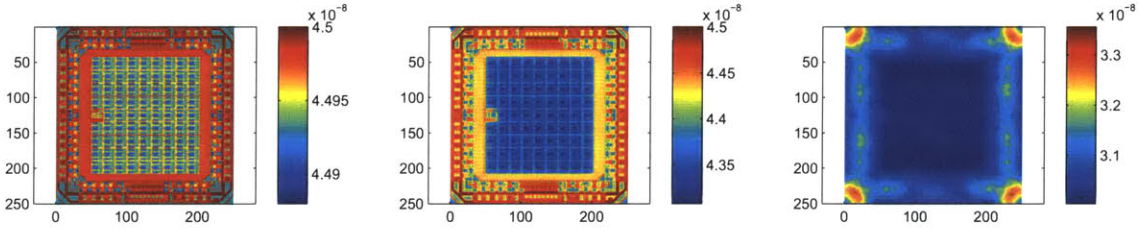
Several barriers prohibit TNIL and UV-NIL from industry-wide adoption. TNIL is a slow process, since it requires the stamp to be held for a significant amount of time



(a) Hierarchical representation of the stamp, with density map on left and feature spacing map (m) on the right.



(b) Pressure distributions (Pa) at times .04 s, 1.1 s, and 4 s.



(c) RLTs (m) at times .04 s, 1.1 s, and 4 s.

Figure 2-2: Example simulation results from `simprint`. The simulations were run on an abstracted chip stamp in hierarchical mode. The imprinted material was modelled as PMMA with a 4.5 nm thickness on a silicon substrate with 1 mm thickness. The hold time was 4 seconds at a temperature of 170° C. The time-steps shown are 0.04 seconds, 1.1 seconds, and 4 seconds.

(seconds) to allow the viscous resist to conform to it. It also requires time to allow the system to heat and cool. UV-NIL overcomes the problem with hold time by using less viscous resists. However, this comes at the cost of air bubbles contaminating the resist as it solidifies. If these air bubbles are too large, they can cause unevenness in etching.

One of the main barriers for NIL is that the alternative, PL, is more developed and is already in use by manufacturers for the length-scales TNIL and UV-NIL can be used for. However, in the future, PL will be limited by the diffraction of light, whereas NIL will not be. Additionally, NIL methods for multilayer stamps have been demonstrated [4]. The ability to imprint multiple layers at once could save several steps in the manufacturing process, and make the increased process time for imprinting more than worth it.

In order for NIL to be viable for industry, the behavior of the imprinting process needs to be more predictable for manufacturers. This requires simulations that can predict the results of an NIL process for any design a manufacturer plans to imprint, including highly complex chip- or even wafer-scale patterns. Thus, hierarchical die- or wafer-scale simulations need to have reliable results. The rest of this thesis studies how hierarchical pattern abstractions affect the accuracy of such simulations.

Chapter 3

Metrics

In this chapter, we develop metrics to quantify how well NIL simulations agree. These metrics are based on the residual layer thickness (RLT). We take the RLT to be the height of the imprinted material at any time during the imprinting process (see figure 3-1). Regions where stamp cavities are being filled will typically have a higher RLT than regions being pressed in by stamp protrusions. The instantaneous RLT is important for manufacturing, because it controls how long the imprinting process must take. Knowing how the RLT evolves with time allows manufacturers to choose when to end the imprinting process, based on specifications or requirements on final RLT. Knowing where on the pattern the RLT deviates from the specifications allows designers to redesign the layout in that region.

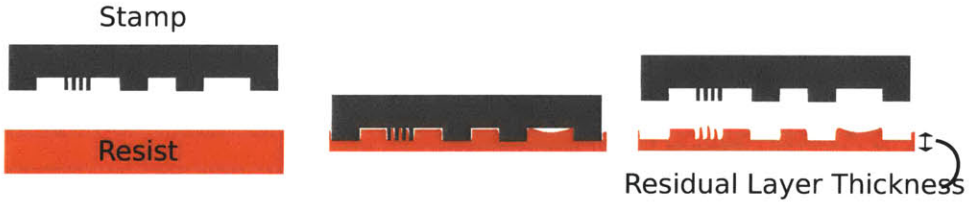


Figure 3-1: The residual layer thickness, illustrated on the diagram from chapter 2 of the imprinting process of NIL.

We first set up a framework for discussing metrics and comparing simulations in section 3.1. We then introduce three types of metrics and explain the motivation for developing them. In section 3.2, we characterize the time evolution of the *difference*

between RLTs, which is key to understanding the metrics. Sections 3.3, 3.4 and 3.5 define the metrics we propose and explore some of their properties. In section 3.6, we link these metrics to measures of error. We reject other metric candidates in section 3.7, and conclude with section 3.8, in which we describe the value of the metrics we chose.

3.1 Models and Metrics

This study focuses on comparing simulations of NIL, but the metrics we develop in this chapter have more general applications. For this reason, we define a *model* of NIL as anything that can produce a time evolution of RLT given a pattern and process parameters. Therefore, a model can be an experiment, a finite element model, or a hierarchical simulation (among other things). Our metrics will measure how close two models are to each other.

3.1.1 Compatible Models

It is useless to compare arbitrary models that can produce arbitrary time evolutions of RLT. For this reason, we restrict ourselves to comparing *compatible* models. Otherwise, we could compare valid models of different processes, and it would be impossible to define a single number that would be useful in comparing these time evolutions.

We define two models as compatible if they are compatible with the same "true" model. In this case, the RLTs they produce have the following properties:

1. *They have the same initial conditions.* The models must match the initial conditions given by the process parameters, so the RLTs are the same at time zero.

2. *They approach the same steady-state.* This is because the stamp will either push the RLT to zero or meet resistance to compression at some minimum value of RLT.

3. *They are reasonable.* The RLTs both follow the RLT produced by the "true" model. The way models deviate from the true model and from each other is described and investigated in the rest of this chapter.

We could define the last item more rigorously. However, defining compatible models rigorously becomes circular as it is informed by the characteristics we observe when studying the error between compatible models (see section 3.2). Therefore, our metrics are based on these observations rather than on a study of the properties of a theoretical framework. However, when we develop our metrics, we will rely on the error behaving reasonably.

3.1.2 Metrics

Having defined the models we are comparing, we now turn to the metrics we use to compare them. The *metric*, ξ, quantifies how similar two models, A and B, are based on the RLT time evolutions the models produce, r_a and r_b. Additionally, the *temporal submetric*, $\xi_t(t)$, quantifies how A and B converge in time, while the *spatial submetric*, $\xi_s(x, y)$, quantifies where they converge on the pattern.

Temporal Submetric. We seek a comparison of the two models, A and B, that varies with time, t, in the simulation. We call this the unscaled temporal submetric, $\xi_t(t)$. It answers the question: *when in time do models agree?* The temporal submetric will indicate at what time in the simulation the two models agree sufficiently.

Spatial Submetric. We would also like a comparison that varies with position in the pattern. We call this the unscaled spatial submetric, $\xi_s(x, y)$. It answers the question: *where on the pattern do models agree?* Using the spatial submetric, we can determine what areas of the pattern the two models disagree on.

Unscaled Metric. The unscaled metric, ξ, will be a single number that quantifies the similarity between the RLTs produced by two models. We seek a metric that can be related to both the temporal and spatial submetrics.

3.2 Characterizing RLT Difference

To compare two models, A and B, we will look at the RLTs they produce, $r_a(x, y, t)$ and $r_b(x, y, t)$, and in particular at the difference between RLTs, $\Delta r(x, y, t) = r_a(x, y, t) - r_b(x, y, t)$. The metrics we wish to develop will quantify the important features of Δr. For these metrics to be useful, we must understand how Δr typically behaves.

Figure 3-2 shows the simulated instantaneous RLT for a test chip imprinted by a typical T-NIL process, as well as sample RLT time evolutions for points on the chip. The time evolutions predicted by two simulations of the same process are plotted together to show how similar they are. They start with the same initial conditions and approach similar steady-states. The differences between the two predictions is most pronounced at the beginning (much of which is cut off in figure 3-2 to make the graphs easier to read).

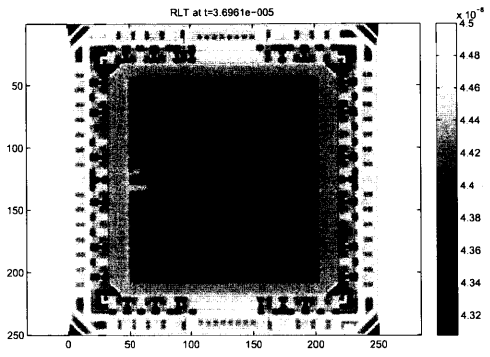
3.2.1 Characteristics of Square Error

For simplicity, and in order to make our metrics symmetric, we ignore the sign of the difference in RLTs produced by models A and B, $(r_a - r_b)$. We instead focus on the square difference of r_a and r_b, $\Delta r^2 = (r_a - r_b)^2$. This is the square error, which allows us to characterize the error without worrying about the sign, and which penalizes larger errors much more than small errors.

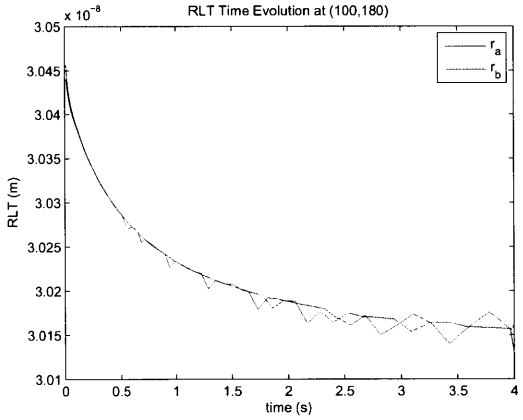
Figure 3-3 shows an idealized square error curve that illustrate the primary features of the time evolution of the square error. There are four of these key features:

1. **Square error starts at the origin.** Because models A and B have the same initial conditions, the square error starts at zero.

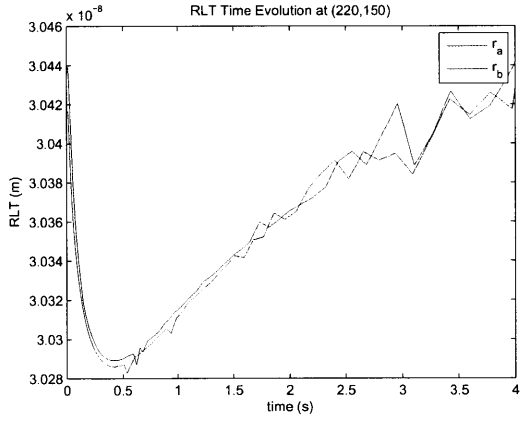
2. **Error peaks quickly near time zero.** The square error peaks as the two models diverge from their initial conditions. We have observed that this peak is usually quite sharp, and occurs early in the time evolution. This is because the simulations we are comparing are particularly sensitive to model assumptions and model parameters near the beginning of the time evolution.



(a) Instantaneous RLT (color scale in meters).



(b) RLT time evolution.



(c) RLT time evolution.

Figure 3-2: The RLT predicted by the simulation of a test chip being imprinted by a typical T-NIL process. The RLT time evolution for two sample points on the pattern is also shown. These time evolutions have been predicted by two slightly different simulations. The time evolutions are cut off in the RLT dimension to make the scale reasonable.

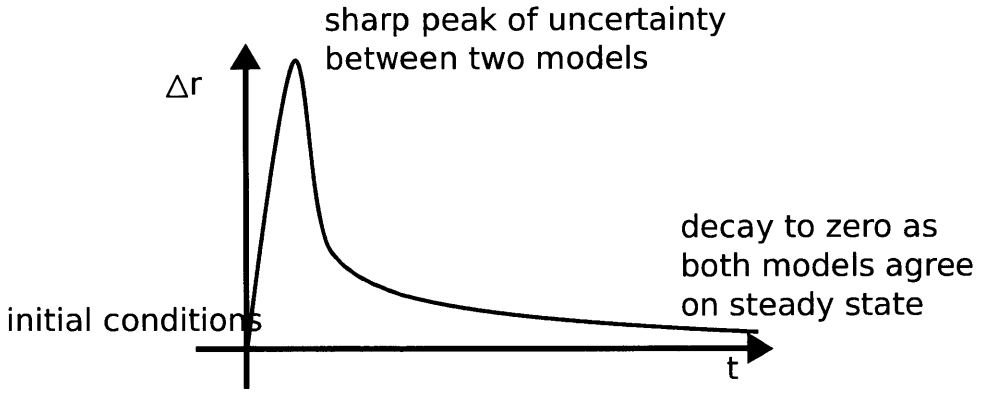


Figure 3-3: An idealized illustration of square error curve. The square error starts at zero since the two models start at the same initial conditions. The two models then diverge quickly before gradually converging. This results in the square error peaking sharply before decaying to zero.

3. **Error has potential zero crossings.** The values for r_a and r_b may cross over each other, and may be equal. Therefore, Δr^2 potentially goes to zero. This can occur before or after the sharp peak, and the peak itself may be masked by such a zero crossing.

4. **Error converges to zero as time increases.** The square error often goes to zero as r_a and r_b approach the same steady state for models that embed similar physical assumptions about final force balance and other NIL model parameters.

3.2.2 Characterizing Noise and Decay

Actual plots of square error may be much harder to interpret than the idealized plots in figure 3-3. Visualizing them on a log-log plot accentuates the features discussed in the previous section, and reveals a more sophisticated way of characterizing the typical evolution of square error as a noisy decay.

Figure 3-4 shows a log-log plot of square error. The characteristic peak and decay are clear, but instead of decaying to zero, the square error appears to become noisy as time increases. In particular, there is a region of decay which captures the way in which two models converge in time, and a second region of noise which captures the

small "steady state" differences between models.

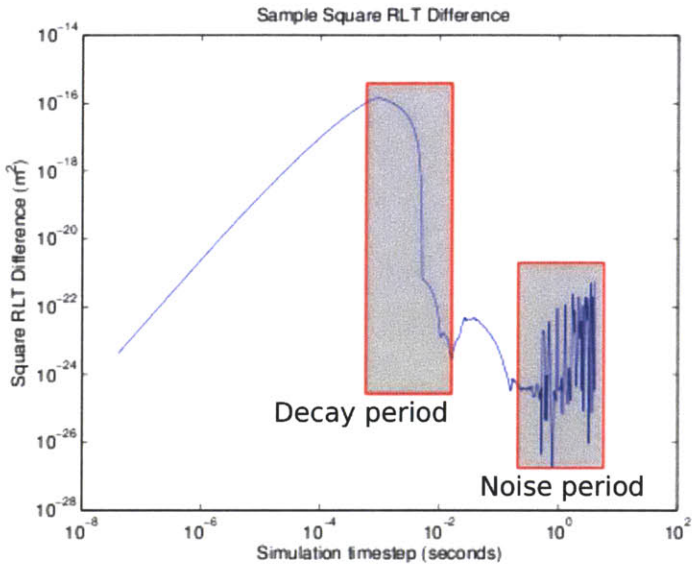


Figure 3-4: A log-log plot of the time-evolution of the square RLT difference. The log-scale in time makes the initial increase and fast "decay period" obvious, but does not give a good sense of how long the "noise period" actually is. The log-scale in square RLT difference illustrates the noisy fluctuations during the "noise period."

We could characterize the square error curves with metrics that describe these characteristics. In particular, we would want to have some metric to characterize the decay region, and a metric to characterize the noise region. In fact, the temporal submetric will usefully characterize the decay region, and the spatial submetric will be helpful for characterizing the noise region. Since the temporal submetric helps summarize how the models differ in time, it will reveal the decay. The spatial submetric will tend to characterize the noise, because it will focus on the long tail of the "steady state."

3.3 Temporal Submetric

We call ξ_t the unscaled temporal submetric, since it has a different value at each point in time. ξ_t is the absolute error's standard deviation (σ_t) calculated across all spatial

simulation points, added to the mean absolute error (μ_t):

$$\xi_t(t) = \sigma_t(t) + \mu_t(t)$$

where:

$$\mu_t(t) = \frac{1}{A_c} \iint |r_a(x,y,t) - r_b(x,y,t)|\, dA$$
$$\sigma_t^2(t) = \frac{1}{A_c} \iint (|r_a(x,y,t) - r_b(x,y,t)| - \mu_t)^2\, dA.$$

A_c is the area over which we integrate (the characteristic area) and r_a and r_b are the RLTs produced by models A and B.

3.3.1 Behavior of Temporal Submetric

The average error (μ_t) between two RLTs is usually small, so the temporal submetric is close to the root mean square (RMS) error across spatial simulation points, as a function of simulation time. Because of this, ξ_t behaves a lot like the square error (with nonlinear vertical compression due to the square root operation) we discussed in section 3.2.1. Figure 3-5 shows several ξ_t curves. These all display the predicted behavior.

In order to study how ξ_t behaves on different pairs of models, we compare simulations that predict the same process with different numbers of time steps. We compare each model to the most fine-grained simulation to produce the curves in figure 3-5. As expected, the error between the models decreases as the number of time steps increases and approaches that of the reference model. The error peaks also shift towards time zero.

3.3.2 Temporal Submetric as Variance

The temporal submetric captures information about the variance of the error between the two models. For the purposes of studying `simprint`, the mean error is typically

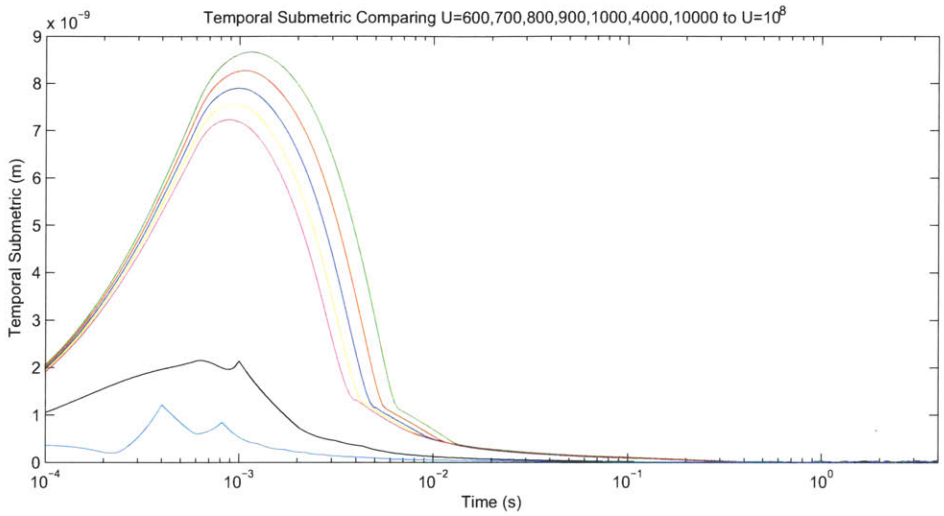


Figure 3-5: The time evolution of the temporal submetric, ξ_t, comparing several simulations to a reference simulation. The curves come from comparing simulations that differ only in the number of time steps simulated. As the number of time steps increases and approaches the reference simulation, the error decreases, and the ξ_t curves shift closer to the origin.

fairly small. Additionally, as time increases, the models typically converge, and the mean approaches zero. Therefore, for later times, ξ_t is simply the standard deviation.

The temporal submetric, $\xi_t(t)$, measures the mean absolute error plus one standard deviation. If we make the approximation that the error across the pattern is normally distributed, then over 68% of the error values across the pattern at time t would be less than $\xi_t(t)$ (and this percentage is higher the farther the mean of the error is from zero). In practice, the error is not in fact normally distributed. However, if the distribution is fairly well-behaved, $\xi_t(t)$ is still a useful measure of the amount of error we can expect.

We can visualize how the errors are distributed by looking at how the different quantiles behave. Figure 3-6 shows plots of the quantiles of the error, as well as histograms of the error at particular times. The histograms demonstrate that the error distributions are well-behaved. The quantile plot of error also shows a sharp peak before the two models converge, as predicted in section 3.2. These characteristics are slightly masked by the mean error fluctuating.

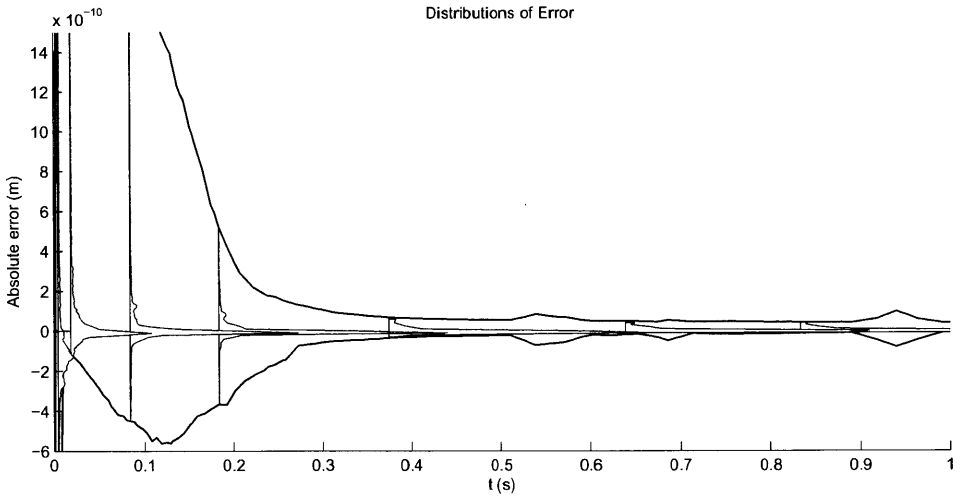


Figure 3-6: A plot of how the range in error values and density of error distribution change with time. The shaded area shows the error values between the 1st and 99th percentiles. Selected error distributions are overlaid on this graph to show how the distribution of error is well-behaved.

The peak and decay characteristics are clearer if we look at a quantile plot of the absolute error. Figure 3-7 shows this, and compares the absolute error quantiles to ξ_t. From this, it is clear that the temporal submetric characterizes the time evolution of the error in a relatively succinct fashion.

3.4 Spatial Submetric

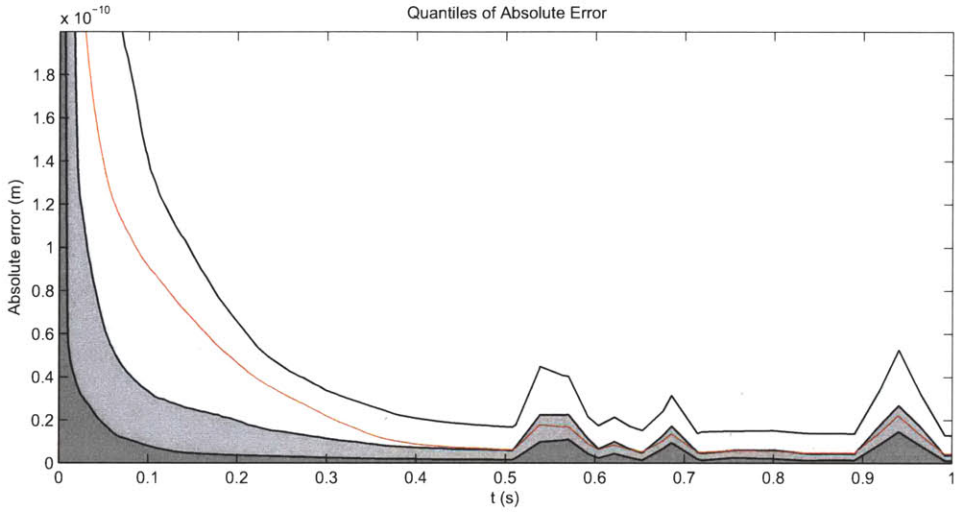
We call ξ_s the unscaled spatial submetric, since it varies with each point in space. We define ξ_s to be the local standard deviation ($\bar{\sigma}_s$) added to the absolute value of the local mean ($\bar{\mu}_s$):

$$\xi_s(x,y) = \bar{\sigma}_s(x,y) + |\bar{\mu}_s(x,y,t_c)|$$

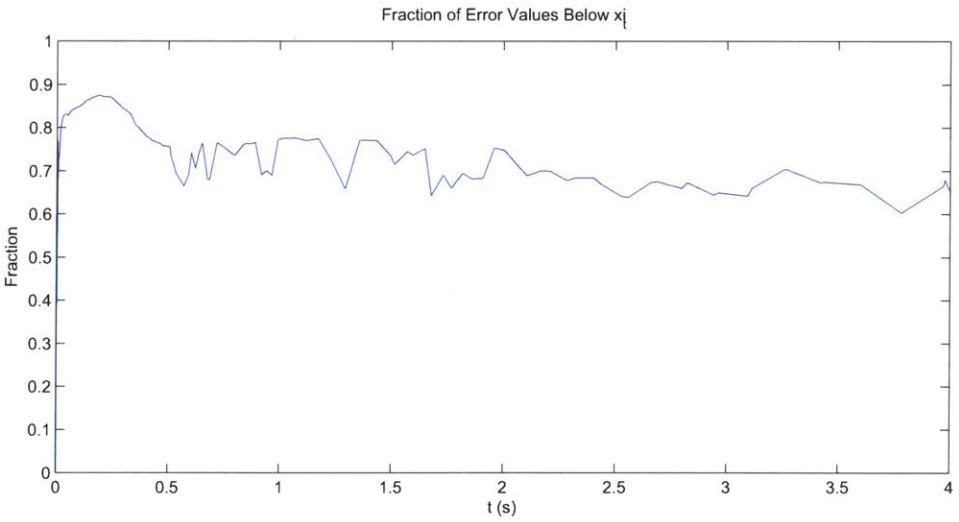
where we define the local standard deviation and mean as:

$$\bar{\mu}_s(x,y,t) = \frac{1}{\Delta t}\int_{t-\frac{\Delta t}{2}}^{\frac{\Delta t}{2}} (r_a(x,y,t) - r_b(x,y,t))dt$$

$$\bar{\sigma}_s^2(x,y) = \frac{1}{t_c}\int_0^{t_c} (r_a(x,y,t) - r_b(x,y,t) - \bar{\mu}_s(x,y,t))^2\, dt.$$



(a) Absolute error quantiles.



(b) Error below ξ_t.

Figure 3-7: Quantile plots of absolute error and ξ_t and the time-varying proportion of error values below ξ_t. The shaded regions show the time evolutions of the 50%, 75%, and 90% quantiles of the absolute error. The temporal submetric is overlaid in red to show that it characterizes the error values. In the lower graph, the proportion of error values that falls below ξ_t is shown as a function of time, demonstrating that ξ_t gives a sense of expected error within fairly tight boundaries.

Here, t_c is the final time we integrate to (the characteristic time), while r_a and r_b are the RLTs produced by models A and B. The metric focuses on fluctuations in the RLT error by combining local measurements of the mean and standard deviation, thereby smoothing the curve. We choose the window size, Δt, to be exponentially increasing with t. By doing this, the quick peak in error is ignored by subtracting out a smoothed version of it using a very tight window. The noisy tail is properly centered by smoothing over a wider range of values in this region.

3.4.1 Behavior of Spatial Submetric

The unscaled spatial submetric characterizes the error values between the models at every point in the pattern. This is useful for determining which parts of the pattern are difficult to simulate, which can indicate the need to simulate more slowly and carefully. Figure 3-8 shows an example of ξ_s. The value of ξ_s at every point is a summary of how the RLT error converged at that point.

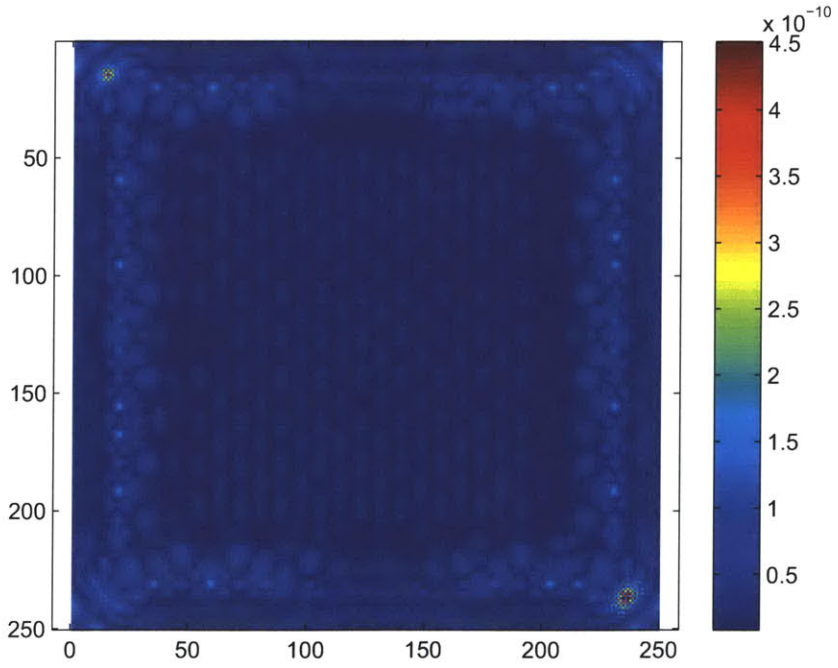


Figure 3-8: Intensity map of the spatial submetric. The value of ξ_s (shown in meters) at every point is a summary of how the RLT error converged at that point.

3.4.2 Spatial Submetric as Variance and Noise

The spatial submetric does a good job of characterizing the noise region of the RLT error discussed in section 3.2.2. Figure 3-9 shows the log-log plots of the error evolution at several points on the pattern, as well as the values of ξ_s^2. This shows that ξ_s indicates the magnitude of the fluctuations during the noise period. It does this by minimizing the effect of the widely varying peak by comparing the variance to a local average.

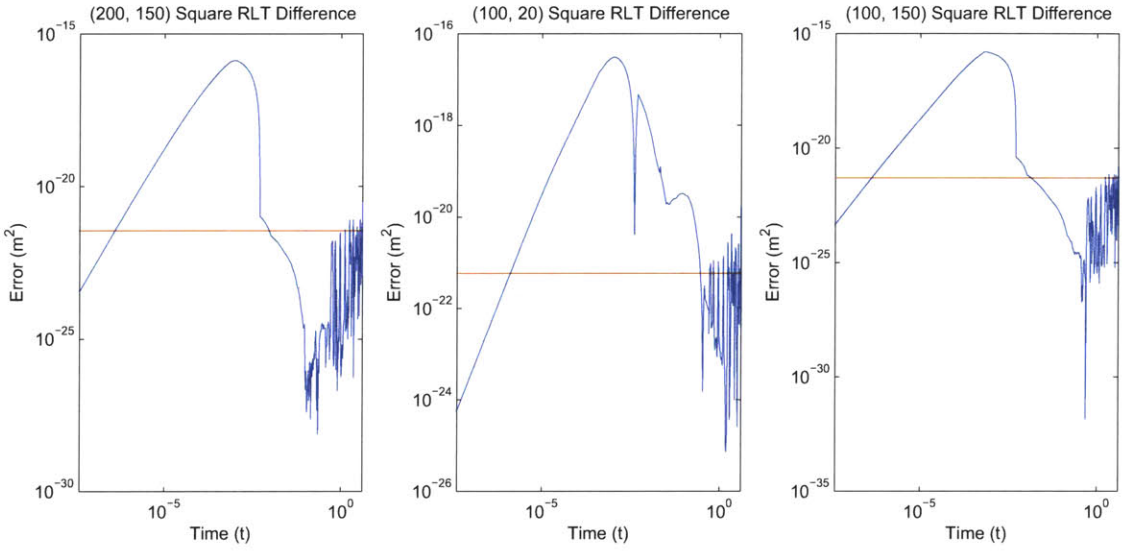


Figure 3-9: Log log plots of the difference in RLT time evolutions at different points throughout the pattern with the corresponding values of ξ_s (horizontal lines) overlaid on the graph. This shows how ξ_s characterizes the "noise region" of the RLT evolution.

We use ξ_s because it captures information about the variance of the error, describing the distribution of the values of error at the corresponding point in the pattern. Typically, a consistent proportion of the error values ($|\Delta r(x, y, t)|$) at some point in the pattern, (x, y), are smaller than the spatial submetric at that point, $\xi_s(x, y)$. Figure 3-10 shows distributions of the error values at several points in the pattern with the associated ξ_s values. The distributions reveal the error values to be peaky with long tails. This is as we might expect, with the tails corresponding to the decay region and the peak corresponding to the noise region. Figure 3-11 shows the distribution of proportions of error values less than $\xi_s(x, y)$ for all points in the pattern. This shows

that ξ_s typically characterizes 85% of the error for this experiment. The error values that are larger than ξ_s usually belong to the early peak in error.

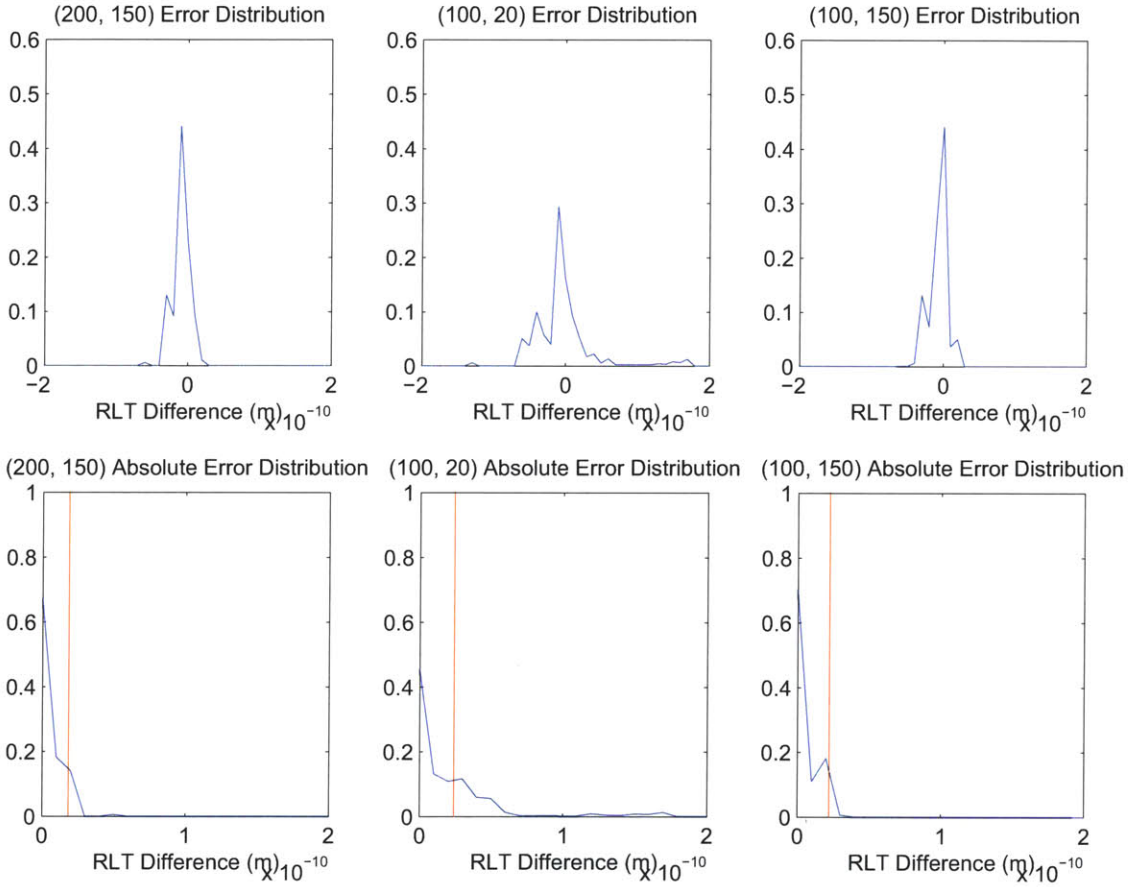


Figure 3-10: The distribution of error values at several points on the pattern. The top row shows the distribution of error values, while the bottom row shows the distribution of absolute error values. Though there are several extremely large values of error, most of the values are relatively small, and the distribution behaves well. The value of ξ_s for each of the points is indicated by a vertical line on the distributions of absolute error on the bottom row. This shows how ξ_s characterizes the error values.

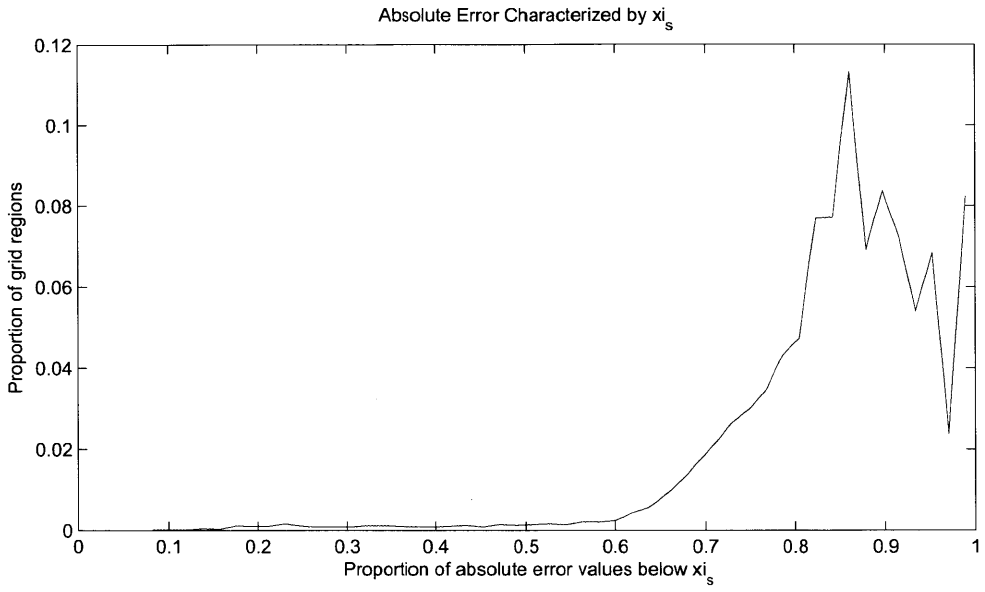


Figure 3-11: A distribution plot of the proportion of error values below $\xi_s(x, y)$ for every grid region (x, y) in the pattern. The peak at 0.85 suggests that ξ_s is larger than 85% of the error values at the corresponding point in the pattern. For the simulations we ran, the noise region was fairly long. For shorter noise regions, we would expect the peak to be at a lower proportion. For longer regions, a higher proportion.

3.5 Unscaled Metric

The unscaled metric, ξ, combines the summaries of the spatial and temporal submetrics into a single number:

$$\xi = \bar{\sigma} + \iint_{A_c} |\bar{\mu}_s(x, y, t_c)| \, dA,$$

where $\bar{\sigma}$ is the local deviation of the error and $\bar{\mu}_s$ is the local mean defined in section 3.4:

$$\bar{\mu}_s(x, y, t) = \frac{1}{\Delta t} \int_{t-\frac{\Delta t}{2}}^{\frac{\Delta t}{2}} (r_a(x, y, t) - r_b(x, y, t))dt$$

$$\bar{\sigma}^2 = \frac{1}{t_c} \int_0^{t_c} \iint (r_a(x, y, t) - r_b(x, y, t) - \bar{\mu}_s(x, y, t))^2 \, dA \, dt.$$

Like the two submetrics, the metric is strongly influenced by the variance of the error. It gives a sense of the scale of the absolute error, and represents the "distance"

between two models. Figure 3-12 shows graphs from figures 3-7 and 3-11 that showed how ξ_s and ξ_t characterize error. This time, a plot of how ξ characterizes the error is overlaid on these graphs. The unscaled metric, ξ, does not do as good a job as the submetrics, but works well considering it is a single number, and therefore is not as detailed a description as ξ_t and ξ_s.

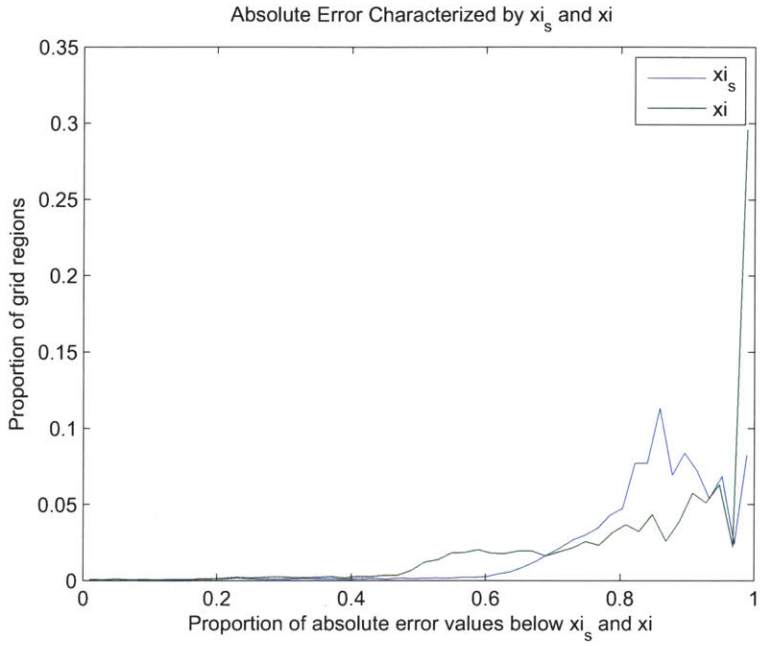
3.6 Metrics and Error

So far, we have referred to all of our metrics as "unscaled." This is because they have units of distance. Each unscaled metric can be scaled to be unitless in several ways. These include dividing by the initial RLT, dividing by the final RLT (or the mean of the RLT at the final time), and dividing by the instantaneous RLT. Doing any of these results in scaled metrics that correspond to the typical percent error between the two models.

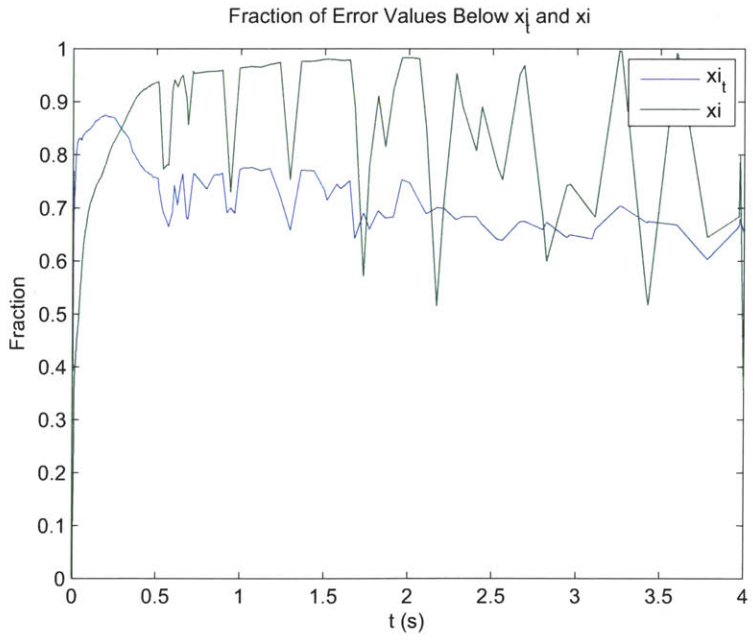
Figure 3-13 demonstrates the use of such scaled metrics, particularly to establish error bounds on simulation outputs such as RLT evolution. The error bars on the graphs correspond to ξ_t at that time. Since ξ_t quantifies the scale of the error at a given point in time, overlaying ξ_t error bars gives us a sense of the range of possible RLTs we might actually observe at that point on the pattern. We could divide ξ_t by the mean RLT at each time to get the percent error.

This gives us measures of error which we call the scaled metrics and scaled submetrics. The scaled temporal submetric, Ξ_t, is scaled by the mean instantaneous RLT, while the scaled spatial submetric, Ξ_s, is scaled by the initial RLT. We also get the scaled metric, Ξ, by scaling the unscaled metric by the initial RLT.

$$\Xi_t(t) = \xi_t(t)/\bar{r}_1(t)$$
$$\Xi_s(x,y) = \xi_s(x,y)/\bar{r}_1(x,y,0)$$
$$\Xi = \xi/\bar{r}_1(0)$$



(a) Error below ξ_s.



(b) Error below ξ_t.

Figure 3-12: Plots showing how the unscaled metric and submetrics characterize error. The unscaled metric characterizes error values unevenly compared to ξ_s, shown by the flatter distribution and the peak at 1. It also characterizes error much less evenly compared to ξ_t, shown by the wider fluctuations. Still, ξ seems to give some sense of the magnitude of typical error values.

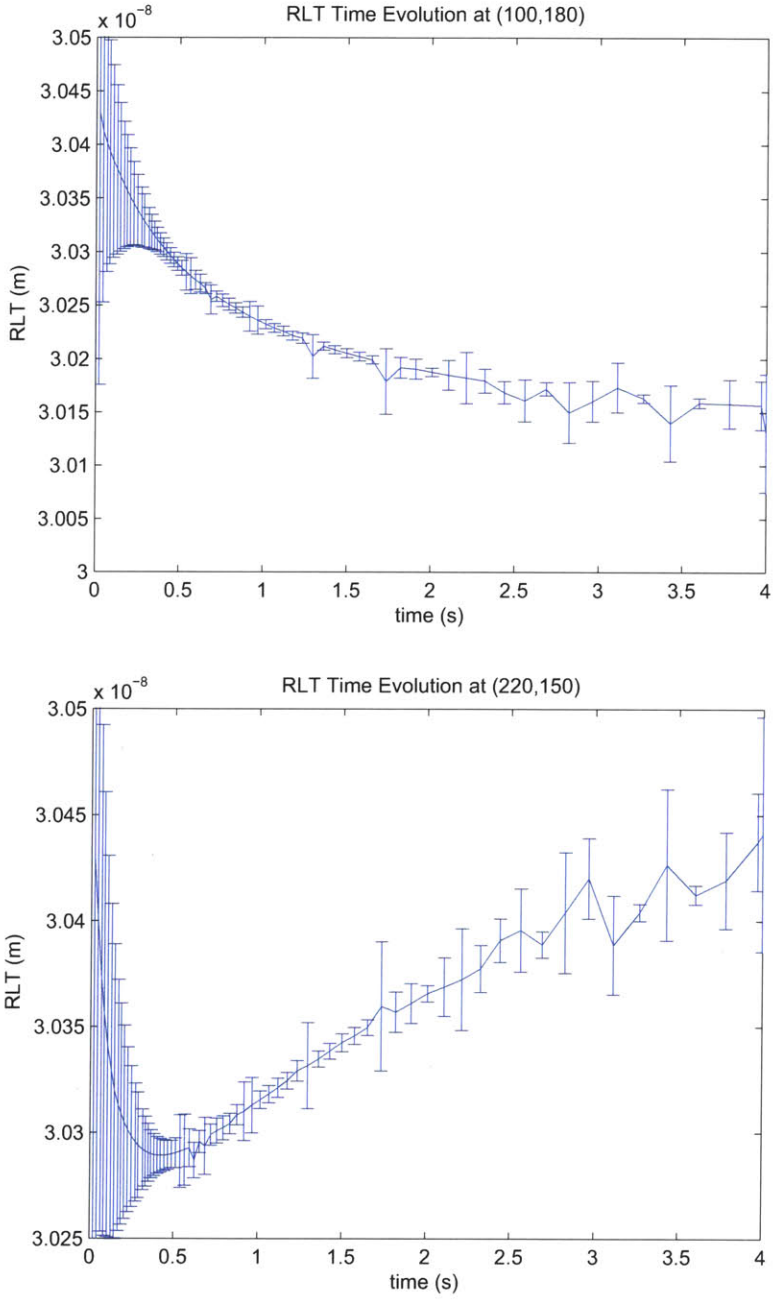


Figure 3-13: Using the metrics as measures of error. The error bars are based on the temporal submetric.

3.7 Other Metric Candidates

As part of our study, we consider several other candidates as metrics. In this section, we briefly review their benefits and drawbacks. We ultimately reject them in favor of the metrics we described in the previous sections.

3.7.1 Temporal Submetric

1. $\xi_t = \frac{1}{A_c} \iint (r_1 - r_2 - \mu)^2 dA + |\mu|$.

   We consider calculating our temporal submetric using the standard deviation and mean of the error instead of the absolute error. This has very similar properties to taking the standard deviation and mean of the absolute error (which we choose to do instead), since the mean error tends towards zero. Thus, this version and the one we defined before act very similarly for later time values. However, we find that our preferred form characterizes the values of error much more effectively for smaller t.

2. $\xi_t = \frac{1}{A_c} \iint (r_1 - r_2)^2 \, dA$.

   The mean square error for every point in time is not much different from the square of the temporal metric we decided on. This is because the average RLT error is typically much smaller than the variance. Since the RLT difference approaches zero in the later simulation steps as the simulations converge, this candidate in fact corresponds to the variance.

   The main issue with this submetric is that we cannot account for cases where the error between the two models have a significant bias.

3.7.2 Spatial Submetric

1. $\xi_s = \frac{1}{t_c} \int_0^{t_c} (r_1 - r_2)^2 \, dt$.

   The mean square error across time does not have a neat interpretation as a useful variance in this form, since the average value of error does not tend toward zero.

2. $\xi_s = \frac{1}{t_c}\int_0^{t_c}(\frac{r_1-r_2}{\bar{r}_1})^2\,dt.$

   This is a weighted mean square error. Since the average RLT tends to decrease, this version of the submetric weights the later steps in the simulation more heavily. This is good because it compares the models after they stabilize. This is bad because the RLT often tends to zero, so the submetric is too sensitive to noise.

3. $\xi_s = \frac{1}{t_c}\int_0^{t_c}\frac{\frac{dR_\Delta}{dt}}{R_\Delta} - \frac{1}{t}\,dt.$

   We propose this submetric on the basis of the model for RLT difference: $R_\Delta = Ate^{-\alpha t}$, where R_Δ is the RLT difference, A is the amplitude that varies with position, t is time, and α is the decay rate that varies with position. In this case, ξ_s is equal to $-\alpha$.

   It is difficult to justify the model used to derive this submetric, though. The t term multiplied in seems like an arbitrary description of the instability at the beginning of the simulation. Additionally, we have no justification for the decay in RLT difference being exponential. In fact, we suspect that the decay is inverse polynomial.

3.8 Conclusion

In this chapter, we developed several metrics for quantifying how similar two models of NIL are. We defined a model very generally as anything that produced the time evolution of an RLT so that we could compare different types of simulations or experiments. In order to make these metrics useful, we looked at the behavior of RLT, and in particular how it varied between two simulations. Thus, the metrics are useful for characterizing the RLT error between models.

The temporal submetric could be very useful for choosing how fine-grained a simulation should be. Given certain necessary bounds on error, one could use the temporal submetric to determine how a set of simulations are converging and choose to stop simulating when the error bounds are appropriate. We are often interested

in the error at later times, since NIL processes tend to continue until the RLT has been reduced appropriately. ξ_t gives useful information in this case, since it shows how error reduces with time.

We can use the spatial submetric to look for problematic types of patterns. From the plots of ξ_s in figure 3-8, it is clear that there is some structure to the values of ξ_s. The areas of pattern with high ξ_s may have commonalities that we can identify.

Finally, the metric, ξ, gives a single number we can use to evaluate the performance of a particular model. This is useful if we are looking to discriminate between models. For example, we might have several different pattern abstractions. We could compare them to an unabstracted model and choose the one with the lowest value of ξ.

Chapter 4

The Effects of Feature Abstraction

In chapter 3, we developed metrics for comparing how two simulation methods differ when simulating the same NIL process. In this chapter, we use these metrics to study the effects of feature abstraction on accuracy in `simprint`. In `simprint`, we can run a simulation on the stamp topography itself (the *flat model*), or we can simulate a low-resolution abstraction of the stamp (the *hierarchical model*). We would like to know how accurate the hierarchical model is. In particular, we have two main research questions:

1. How closely do different hierarchical approximations follow the flat model?

2. What features of the stamp topography should we track in the hierarchical model?

Hierarchical modeling abstracts a stamp topography by approximating patches of complex topographies as patches of a regular pattern. Without hierarchical modeling, simulating the imprinting of a full chip is currently not tractable. Determining how close hierarchical approximations are to the flat model will allow us to bound error on `simprint`'s hierarchical modeling method. It will also help guide our decision about what hierarchical level a stamp should be simulated on. Determining which topography features need to be accounted for in a hierarchical model will guide decisions on what kinds of regular patterns to include descriptions of in a hierarchical simulation.

We explain what hierarchical modeling is and how `simprint`'s hierarchical mode represents stamps in section 4.1. We then describe how we calculate our metrics in a discrete space where time-steps and spatial values may not align in section 4.2. Section 4.3 explains how we will answer our research questions and attempt to improve the hierarchical model. We conclude by presenting our results in section 4.4 and discussing them in section 4.5.

4.1 Background

In `simprint`, we can carefully simulate at the flat level or more quickly at the hierarchical level. Even if the hierarchical approximation describes the topography exactly, though, some of the RLT variation within each cell will be lost. However, the RLT variation across the pattern tends to be much more significant than the variation within a small patch, and we can increase the granularity of our abstraction if that is not the case.

We test a very simple topography abstraction scheme that approximates square cells as patches of parallel lines. Each cell has a density and a feature scale value, from which the regular pattern can be inferred. Cells in `simprint` can also be approximated as a regular array of square holes and square protrusions [14], but here and in chapter 3, we only use parallel lines. To abstract a topography in this way, we simply extract the density and perimeter. The feature scale is calculated by combining the two values with the assumption that the patch is a regular pattern of parallel lines.

4.2 Extending Metrics to the Discrete Space

We now wish to apply our metrics from chapter 3 to study the error of this hierarchical method. In chapter 3, we defined our metrics assuming the RLT time evolution was a continuous function. However, we typically apply these metrics to discrete functions. Packages like `simprint` give us the RLT values for discrete time-steps on a discrete spatial grid. Often, the two simulations we are comparing are valued at different

time-steps or have different grid discretizations. In these cases, we have a number of options for extending our metrics.

Conceptually, the approach we choose is to linearly interpolate between time-steps and assume that these RLT values hold for all values within a cell on the discrete grid. With these ideas, we can translate the continuous formulation of our metrics. We produce values of ξ_t for every time-step in either simulation, and we produce values of ξ_s on a grid with a discretization length that is the lowest common multiple of the two simulations' discretization lengths.

4.3 Method

We plan to look for correlations between hierarchical simulations and error. We have two primary goals: measuring the effect of abstraction level on accuracy and finding features of stamp topologies that correlate with error.

We calculate several features from the flat topography, because we want to identify which features will be most valuable to track. In particular, we extract the following features in each cell:

1. Density.
2. X-perimeter.
3. Y-perimeter.
4. Number of shapes along the vertical direction.
5. Number of shapes along the horizontal direction.

The x- and y-perimeter values are found by projecting the horizontal or vertical lines of shapes within a cell and adding them. The number of shapes in the vertical and horizontal directions are found by counting the maximum number of shapes intercepted by a horizontal or vertical line.

In this chapter, we simulate several test patterns using `simprint`'s flat and hierarchical modes. To cover a range of geometries, we generate random patchwork stamps

with varying features. Each patchwork stamp is then simulated on the flat level and on several hierarchical levels. Based on these simulations, we will look at how ξ_t and ξ_s vary with abstraction level, and how ξ_s varies with several stamp topography features.

4.3.1 Test Stamps

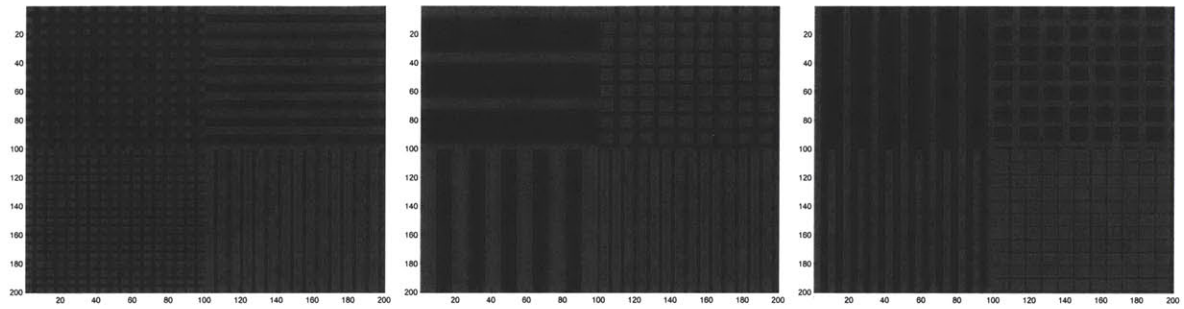
The patchwork stamps are 2×2 grids of 4 patches, where each patch has a different regular pattern (square holes, square protrusions, parallel lines) with different feature sizes. In order to encourage variation in RLT, we fix the densities for each patch so that we have two 50% density patches on the diagonal, one 25% density patch, and one 75% density patch. Each patch is 10 microns long, with a discretization of 100 nm – comparable to the chip we studied in chapter 3 (10 micron-long hierarchical regions, 120 nm features). Figure 4-1 shows sample patterns.

4.3.2 Simulations

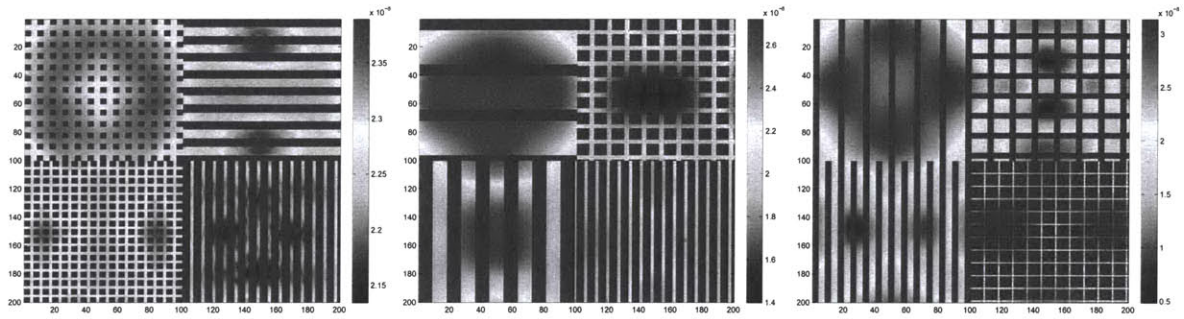
We run hierarchical simulations at the flat level, and at the $10\times$, $20\times$ and $50\times$ levels. Though it could be useful to run a hierarchical simulation at $100\times$ (the patch size), we encountered a problem with running stamps smaller than 3×3. Additionally, we will see very little change in the results of the hierarchical simulations, so compensating for this issue is unnecessary.

4.4 Results

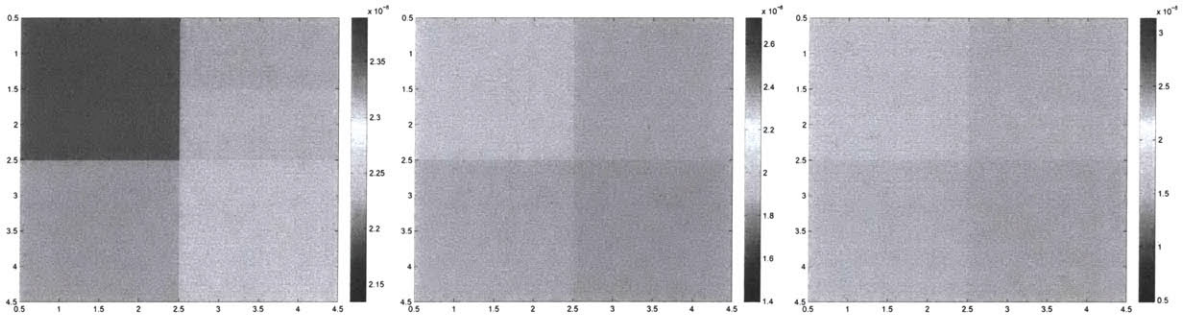
We ran 40 replicates of 2×2 patchwork patterns at the flat level and three abstraction levels. We tried to mimic the simulation conditions of the chip from chapter 3, but we reduced the hold time to 1 second (from 4). The final RLTs predicted by the flat model and the $50\times$ hierarchical model are shown in figure 4-1



(a) Pattern.



(b) Flat.



(c) Hierarchical.

Figure 4-1: Test patterns and the RLTs predicted by flat and hierarchical models. All of the patterns have similar densities in similar patches in order to promote RLT variation. Despite this, the RLT variation is minimal, as the scale of the final RLT predicted by the flat simulation and by the 10× hierarchical simulation show (note the scale on the color legend).

4.4.1 Effect of Abstraction Level on Accuracy

Our first research question (how closely do different hierarchical approximations follow the flat model?) can be divided into two parts. We first need to determine what error we should expect in transitioning from the flat model to the hierarchical model. Second, we need to see how accuracy changes as we change the abstraction level of the hierarchical model.

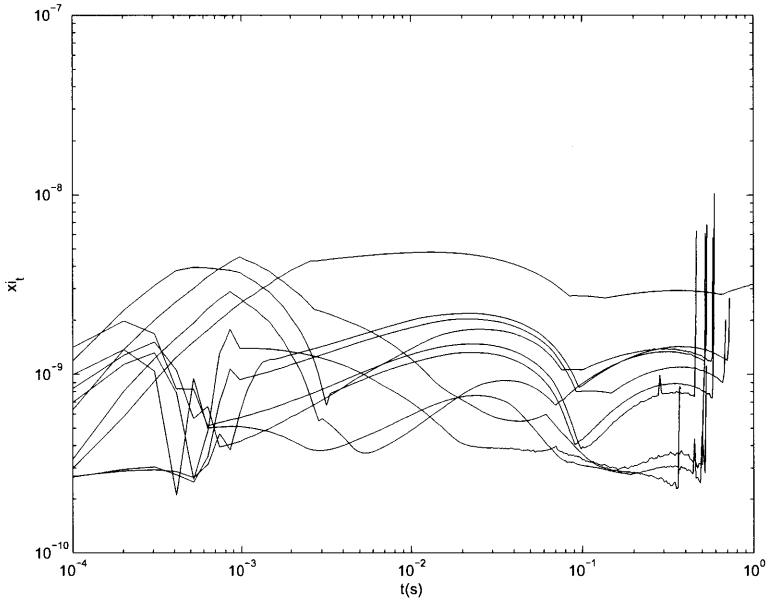
To determine the error we should expect between the flat model and the hierarchical model, it is useful to look at ξ_t. For each stamp we simulated, we have a ξ_t time-evolution curve. Since the temporal submetric tends to be dominated by the standard deviation, we combine the curves by taking the square root of the average of their squares. Figure 4-2 shows the range of ξ_t curves and the ξ_t curve produced by combining the metric time-evolutions for all of the patches.

Based on the combined ξ_t curve, we can see that error tends to stay around 10^{-9} m. Compared to the relief size of 4.5×10^{-8} m, this error is around 2.2%. This is a very reasonable amount of error, especially given the gains in performance.

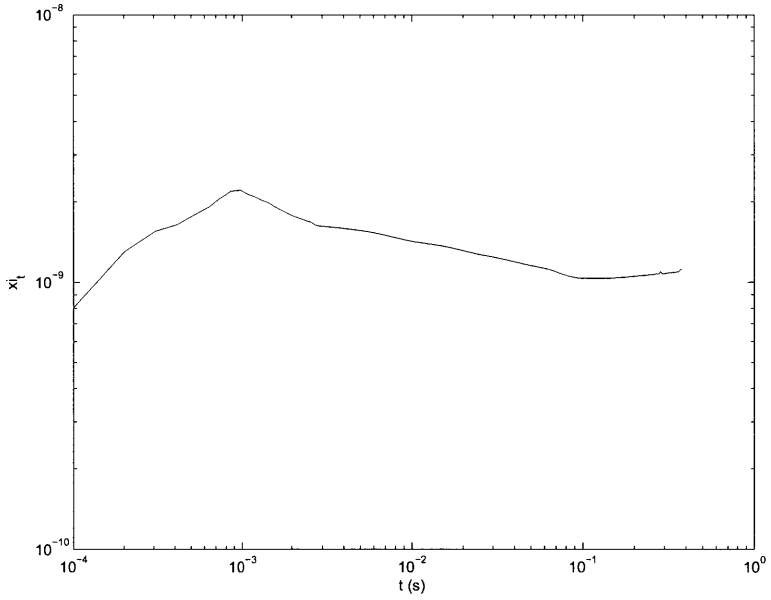
To analyze how accuracy is affected by abstraction level, we look at ξ_s. We simulated over three abstraction levels, which gives us three measurements of ξ_s for every patch. Figure 4-3 shows these ξ_s values plotted against the abstraction level. Patches with smaller error at the $10\times$ level generally appear to increase in error at the $20\times$ and $50\times$ levels. However, this rise in error is not significant, and can be explained by the increased resolution in the lower abstraction levels. Overall, there does not appear to be any trend, except that ξ_s is very similar across abstraction levels. From this, we conclude that abstraction level mainly affects the resolution of RLT results. The main source of error appears to be the transition from the flat to the hierarchical model.

4.4.2 Effect of Features on Accuracy

The 10 random 2×2 patchwork patterns gave results for 40 patches with varying features. This allows us to look for correlations between features and error. Each of



(a) ξ_t curves.



(b) Combined ξ_t curve.

Figure 4-2: The ξ_t curves produced by the 50× hierarchical simulations. In the first graph, all of the ξ_t curves for the 10 simulations are shown. In the second graph, the ξ_t curves have been combined.

the cells in the flat simulation produce a different value of ξ_s, so for each patch, we combine ξ_s values by taking the average within each patch. (We also tried taking the square root of the mean of ξ_s^2, since ξ_s acts like a deviation.)

We calculate the following features, which can be derived from the features listed in section 4.3:

1. Density.
2. Perimeter.
3. Maximum x/y-perimeter ratio.
4. Number of polygons.
5. Average feature size.
6. Maximum feature size.
7. Minimum feature size.

The graphs of some of these features and ξ_s are shown in figures 4-4 through 4-8.

4.5 Discussion

We have two reasons for running several pattern simulations at both the flat and hierarchical levels. The first is to determine the error we should expect between the flat model and the hierarchical models of different abstraction levels. Our conclusions are illustrated by the previously mentioned figures 4-2 and 4-3, which show the ξ_t time-evolution and the effect of abstraction level on ξ_s. These results indicate that the hierarchical model is a good approximation of the flat model, but that most of the error between the flat model and the hierarchical models comes from converting a flat topography to a hierarchical representation. The abstraction level does not appear to have a significant impact on the error.

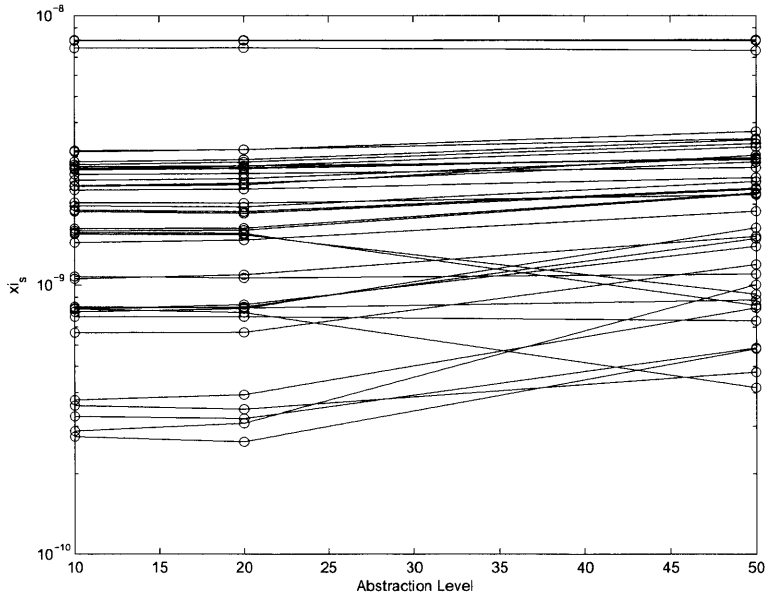


Figure 4-3: ξ_s for each patch plotted against the abstraction level of the hierarchical simulation.

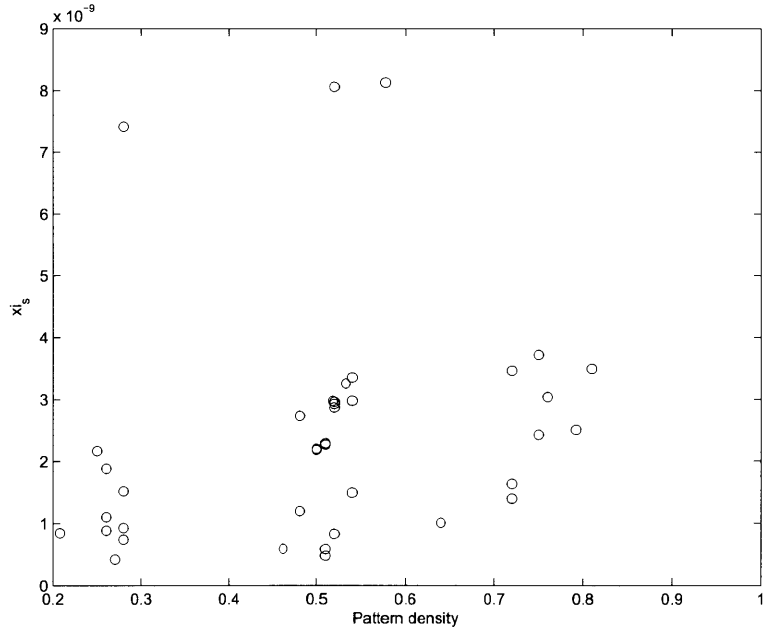


Figure 4-4: Effect of density on spatial error metric.

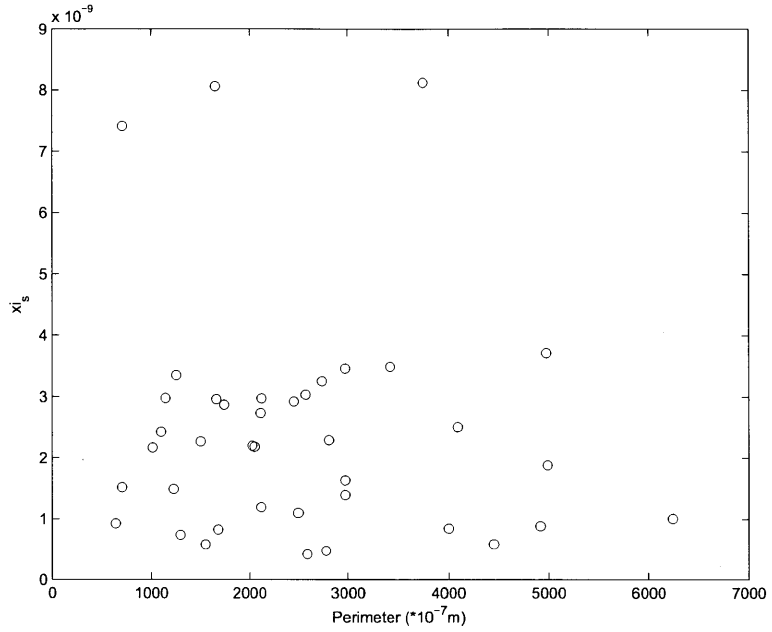


Figure 4-5: Effect of perimeter on spatial error metric.

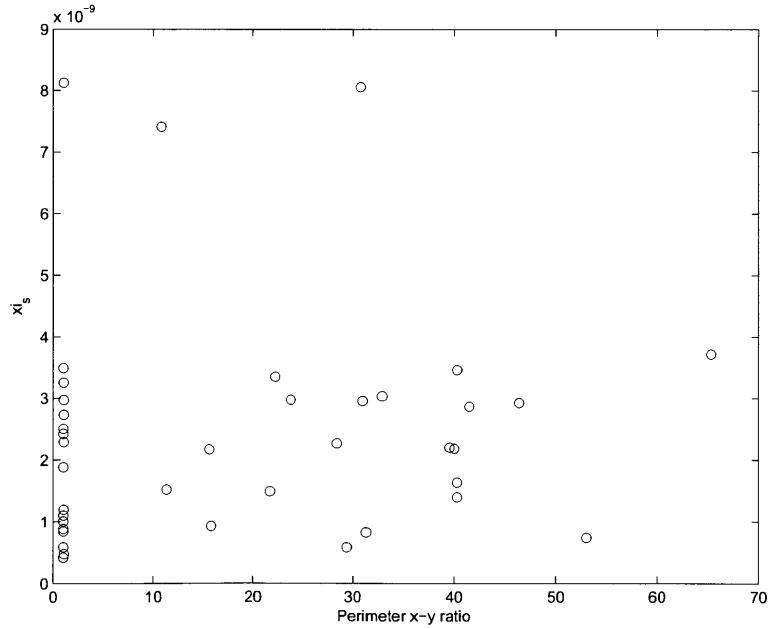


Figure 4-6: Effect of perimeter ratio on spatial error metric.

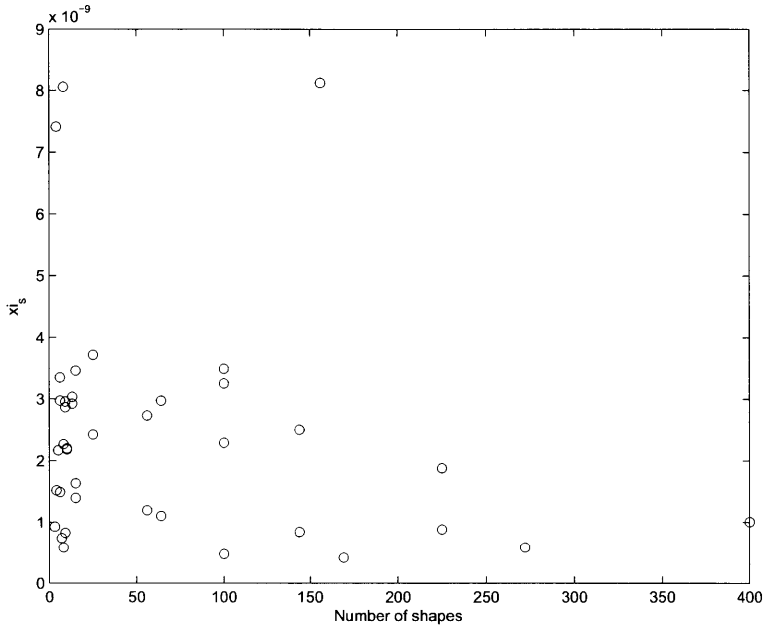


Figure 4-7: Effect of number of polygons on spatial error metric.

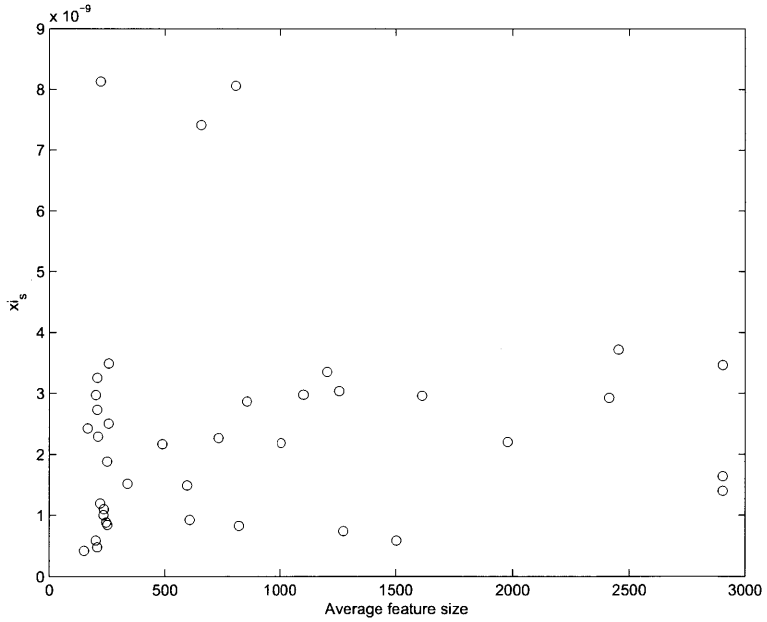
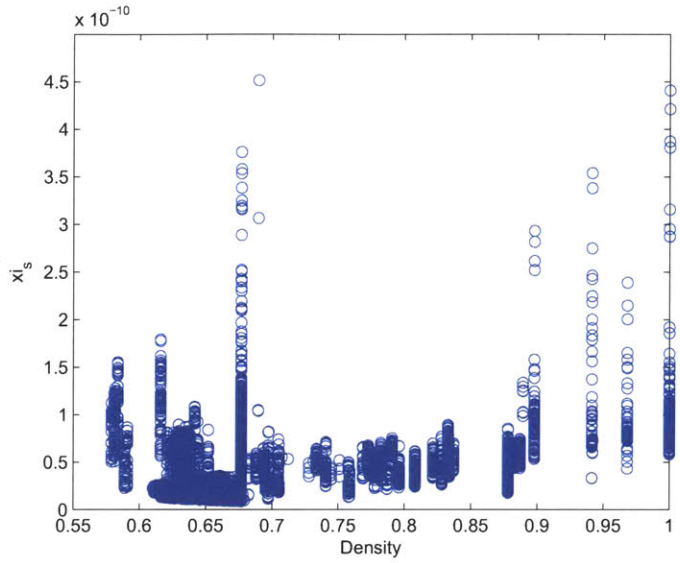


Figure 4-8: Effect of feature size on spatial error metric.

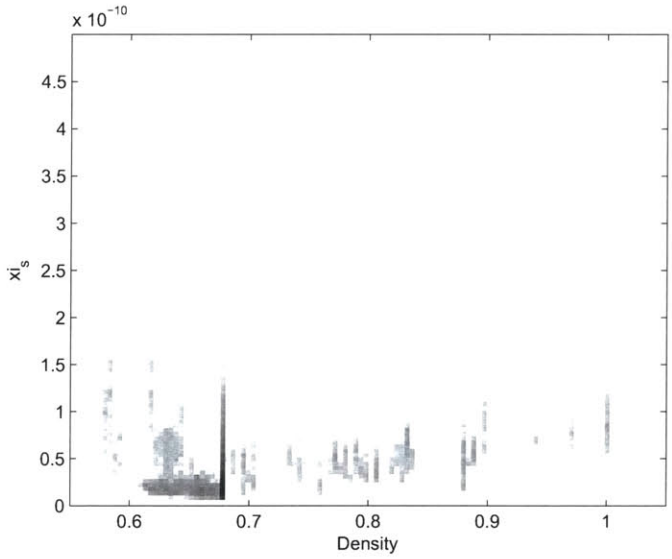
Our second reason for running these simulations is to recommend topography features to track in the hierarchical model. However, none of the features we track show a clear correlation with the error.

One possible reason for this lack of correlations is that the interactions between patches of different feature scales is more important than the intra-patch interactions themselves. To address this possibility, we could increase variability within each patch. We could do this by changing the feature size gradually across a patch, using lines of different widths or squares and rectangles of different sizes. This would also allow us to test features such as standard deviation. Alternatively, we could simulate several 2×2 stamps where one of the patches varied while the other three were identical and remained the same.

A second possible reason for the lack of correlations is that the patterns we tested are too regular. In chapter 3, we noticed structure in the ξ_s values that compared simulations of chip patterns with different time-step sizes (figure 3-8). Figure 4-9 shows how these values correlate with pattern density. It indicates that high and low pattern density areas are more sensitive to time-step size. We expect similarly interesting correlations in ξ_s values comparing flat and hierarchical simulations, and could simulate patches of topography from a real chip layer. This would give us irregular patterns and perhaps reveal more about how approximating each cell as a regular array impacts error. Additionally, such a study would allow us to comment on how hierarchical simulations correlate to flat simulations in more "real" conditions.



(a) Correlation of error to density.



(b) Histogram plot.

Figure 4-9: Sensitivity of pattern density to time-step size. The ξ_s values from figure 3-8 (which compared simulations with different time-step sizes) are plotted against the pattern density of the chip that was being simulated. The second plot shows a histogram plot of the first one, to give some idea of how many points are being plotted in each region. This second plot reveals that higher and lower pattern density areas are more sensitive to time-step size.

Chapter 5

Conclusion

Developing a reliable hierarchical simulation method is critical for industry adoption of NIL. Without this, the cost of iterating on NIL stamps will be prohibitively expensive, and NIL use will be limited to regular pattern arrays. The currently accepted lithographic technique for this length scale is optical or photolithography (PL). However, PL is approaching its physical limit due to the properties of diffraction [2, 1, 6, 14]. Additionally, NIL can imprint multiple layers at a time, potentially saving several manufacturing steps [4]. Thus, promoting NIL use in industry could result in gains in process capabilities and throughput.

The goal of our research was to aid in the development of reliable hierarchical simulation methods of NIL. In particular, we focused on two related problems:

1. How to evaluate the reliability of a hierarchical simulation.

2. How to improve the accuracy of hierarchical simulations.

In chapter 3, we developed metrics for comparing models of NIL. In chapter 4, we used these metrics to evaluate the reliability of `simprint`'s hierarchical mode and search for features to improve this reliability.

5.1 Metrics

One of the main contributions of this thesis is a set of comparison metrics for models of NIL. With these metrics, we can now study how closely different models of TNIL match each other. Additionally, we can measure how model closeness varies with simulation time value and pattern region.

5.1.1 Applications to Research

We developed these metrics to be general-purpose, in the hope that they could be used in many areas of TNIL research. In this thesis, we use them to study the similarity between different modes of simulation in `simprint`. We could also use them to compare different simulation methods and models. Additionally, one could adapt them slightly to measure error between models and experiments.

5.1.2 Applications to Industry

Our metrics could also be used to guide simulation parameter choices. In simulations, there is often a trade-off between simulation accuracy and speed. The speed of a simulation can be increased by decreasing the granularity (lowering the number of time steps or abstracting the feature map), but it is difficult to tell what the effect of these optimizations will be on accuracy. With our metrics, we can now run several very fast simulations and gauge the accuracy bounds by comparing the fast simulations to each other. We can continue to increase the accuracy and use the metrics to determine when we have achieved acceptable bounds on accuracy.

5.2 Metric Limitations

In this thesis, we focused entirely on evaluating models of TNIL. As we pointed out in chapter 2, the requirements and conditions of UV-NIL are very different than those of TNIL. Thus, using the RLT time-evolution as a basis for UV-NIL metrics would

probably not be sufficient. However, many of the ideas we used in developing metrics for TNIL would be applicable to developing metrics for UV-NIL.

Additionally, we based our experiments entirely on one simulation package. Though we have a very good idea of how our metrics behave with `simprint`, they might not be as well-behaved for another model. We built the metrics on assumptions about NIL models that we believed were clear and reasonable. However, we also incorporated the observations we gathered from our experiments, in addition to using those experiments to confirm that the metrics we derived were reasonable.

5.3 Simprint's Reliability in Hierarchical Mode

In chapter 4, we studied the reliability of `simprint`'s hierarchical mode in detail. Our main result was evidence that `simprint`'s hierarchical mode is a good approximation of the analytical model it is based on. Based on our metrics, we estimated that the hierarchical model agrees with the flat model within on the order of 2.2%. We also concluded that hierarchical abstraction level does not have a significant effect on accuracy. This indicates that abstraction level should be chosen primarily based on simulation time and memory considerations as well as some idea of the length-scale of RLT variation, rather than choosing based on how hierarchical abstraction degrades accuracy.

5.4 Hierarchical Pattern Exploration

Hierarchical simulations rely on hierarchical representations of the patterns they are modeling. In `simprint`, feature-rich patches are approximated as regular arrays of shapes. For example, complex mazes of trenches with cuts in them might be approximated as parallel lines. A series of irregular rectangles might be approximated as square holes. Deciding what regular array of shapes a particular pattern region is best approximated by is very important for the accuracy of `simprint`'s hierarchical mode. Our metrics provide a way to improve hierarchical simulations by searching

for better pattern descriptors.

We used our metric to look for straight-forward correlations between features of a stamp topography and errors when modeling that stamp. Unfortunately, we found no correlation using the approach from chapter 4. However, the idea of searching for pattern descriptors using our metrics still seems valid. We suspect that the lack of correlations found in chapter 4 was due to the simplicity of the features we were tracking, or to the simplicity of the patterns we were simulating.

Our metrics open options for methodically improving hierarchical simulations. We could simulate patterns derived from real stamp designs at the feature level and several abstraction levels, and look for correlation between the error and first order features (such as density and perimeter within a single patch) or second order features (such as difference in densities between two patches or the ratio of perimeters between two patches). Alternatively, we could try abstracting real topographies as different regular patterns and evaluate how good these abstraction methods are based on our metrics.

This kind of study could be even more valuable on a hierarchical model of UV-NIL. TNIL resists are very viscous compared to the liquid UV-NIL resists. In TNIL, resist moves relatively slowly from regions of high pattern density to regions of low pattern density. The shape of the patch probably has less impact in this case than it would if the resist were flowing more quickly, as in the case of UV-NIL. In both TNIL and UV-NIL, selecting good pattern descriptions for hierarchical modeling and understanding the effects of pattern geometries is very important for accurate hierarchical simulations.

Bibliography

[1] Se Hyun Ahn and L. Jay Guo. Large-area roll-to-roll and roll-to-plate nanoimprint lithography: A step toward high-throughput application of continuous nanoimprinting. *American Chemical Society Nano*, 3(8), August 2009.

[2] Tobias Balla, S. Mark Spearing, and Andrew Monk. An assessment of the process capabilities of nanoimprint lithography. *Journal of Physics D: Applied Physics*, 41, 2008.

[3] Elizabeth A. Costner, Michael W. Lin, Wei-Lun Jen, and C. Grant Wilson. Nanoimprint lithography materials development for semiconductor device fabrcation. *Annual Reviews of Materials Research*, 39:155–180, 2009.

[4] L. Jay Guo. Nanoimprint lithography: Methods and material requirements. *Advanced Materials*, 19, 2007.

[5] Yoshihiko Hirai. UV-nanoimprint lithography (NIL) process simulation. *Journal of Photopolymer Science and Technology*, 23(1):25–32, June 2010.

[6] Ji-Hoon Kang, Kwang-Seop Kim, and Kyung-Woong Kim. Molecular dynamics study on the effects of stamp shape, adhesive energy, and temperature on the nanoimprint lithography process. *Applied Surface Science*, 257:1562–1572, 2010.

[7] Nikolaos Kehagias, Vincent Reboud, Clivia M. Sotomayor Torres, Vadim Sirotkin, Alexander Svintsov, and Sergey Zaitsev. Residual layer thickness in nanoimprint: Experiments and coarse-grain simulation. *Microelectronic Engineering*, 85, January 2008.

[8] Xiaogan Liang, Hua Tan, Zengli Fu, and Stephen Y. Chou. Air bubble formation and dissolution in dispensing nanoimprint lithography. *Nanotechnology*, 18(2):025303, 2007.

[9] Takashi Nogi and Takahisa Kato. Influence of a hard surface layer on the limit of elastic contact–part I: Analysis using a real surface model. *Journal of Tribology*, 119:493–500, July 1997.

[10] Shravanthi Reddy and Roger T. Bonnecaze. Simulation of fluid flow in the step and flash imprint lithography process. *Microelectronic Engineering*, 82, July 2005.

[11] Harry D. Rowland, Amy C. Sun, P. Randy Schunk, and William P. King. Simulations of nonuniform embossing: The effect of asymmetric neighbor cavities on polymer flow during nanoimprint lithography. *Journal of Vacuum Science and Technology B*, 23(6), November 2005.

[12] Helmut Schift. Nanoimprint lithography: An old story in modern times? A review. *Journal of Vacuum Science and Technology B*, 26(2):458–480, March 2008.

[13] Vadim Sirotkin, Alexander Svintsov, and Sergey Zaitsev. Coarse-grain simulation of viscous flow and stamp deformation in nanoimprint. *Journal of Vacuum Science Technology B*, 25(6), November 2007.

[14] Hayden K. Taylor. *Modeling and Controlling Topographical Nonuniformity in Thermoplastic Micro- and Nano-embossing.* PhD thesis, Massachusetts Institute of Technology, September 2009.